Université de Liège
Faculté des Sciences Appliquées

Integration of ATM under TCP/IP to provide services with minimum guaranteed bandwidth

Thèse présentée en vue de l'obtention du grade de Docteur en Sciences Appliquées par

Olivier Bonaventure
Ingénieur Civil Electricien (Informatique)

Octobre 1998

A la mémoire de Brigitte

Acknowledgements

I would first like to express my gratitude to Prof. André Danthine who allowed me to work on this thesis in the Research Unit in Networking at the University of Liège. He helped to shape the contents of this thesis by providing feedback and allowing me to participate in interesting research projects and encouraged me to finalize this work. Prof. Guy Leduc also played a key role in this thesis by always accepting to discuss on technical problems and by providing many important comments on several drafts of this thesis. Dr. Guido Petit allowed me to complete this thesis by encouraging me to continue to work on this topic during the year I spent in the Traffic Technology group at the Alcatel Alsthom Corporate Research Center in Antwerp. Prof. Jon Crowcroft and Prof. Jean-Yves Le Boudec have also contributed to the finalization of this thesis by providing useful feedback and encouragement to pursue this work.

I would also like to thank Prof. André Danthine, Prof. Guy Leduc, Prof. Jon Crowcroft, Prof. Jean-Yves Le Boudec, Dr. Guido Petit, Prof. Marc Van Droogenbroeck and Prof. Pierre Wolper for having kindly accepted to review this thesis.

Many people have helped me for parts of this thesis. The first one is Espen Klovning, with whom I had the pleasure to collaborate during two years. The measurements described in chapters 4 and 6 are one of the results of this collaboration. The simulations described in this thesis would probably have never been done without the help of Prof. Jean-Yves Le Boudec and Sam Manthorpe who kindly allowed me to reuse the STCP simulation tool developed at Ecole Polytechnique Fédérale de Lausanne (EPFL). The measurements discussed in chapter 4 were performed with the help of Jocelyne Lemagnen, Olivier Danthine and Pierre Boyer. Juha Heinanen and Roch Guérin have provided useful feedback on the first part of chapter 7.

I would also like to thank all my colleagues at University of Liège for the nice working atmosphere and in particular Laurent Mathy and Clarence Filsfils for many interesting discussions. In Antwerp, my work has benefited from many fruitful discussions with Jordi Nelissen, Emmanuel Desmet, Johan Witters, Peter Barri, Michel Henrion, Stefaan De Cnodder and Johan Verkinderen.

Finally, my greatest thanks to Joëlle for her love and unlimited support, to Simon for the countless efforts he made to disturb me while I was writing this thesis and to Laurine who reminded me of the final deadline.

My work has been partially supported by the European Commission in the framework of the RACE 2060 CIO, RACE M1005 ACCOPI and the ACTS AC051 OKAPI programmes and the Flemish Institute for the promotion of Scientific and Technological Research in Industry (IWT). The measurements discussed in chapter 4 were partially supported by Belgacom.

Contents

Chapter 1

Introduction

1.1 The evolution of the data networks

Introduced in the early fifties to support applications such as airline reservations, the data networks have rapidly evolved since then and especially during the last twenty years. During the sixties and the early seventies, the data networks were mainly used to attach devices such as terminals or printers to large mainframes or to directly connect two mainframes together. Most of these "networks" were built with serial or leased lines. From a protocol point of view, "ad-hoc" solutions and proprietary protocols were the rule. The ARPANET protocol suite was one of these proprietary protocols.

Several important changes occurred during the seventies. The first change is the introduction and the "proliferation" of minicomputers such as the Dec PDP series. Until then, the computers were very expensive and few companies could afford to buy several computers. Suddenly, it was possible to have several minicomputers at the same location or even in the same department. This quickly resulted in a need to efficiently interconnect these computers. This need was one of the motivations for the Local Area Networks (LAN)[1] which were developed at that time. The bandwidth available on these LANs (a few Mbps) appeared to be "unlimited" for the computer scientists used to low speed serial lines. An important characteristic of these LANs was that the available computers, due to their limited processing power, were only able to utilize a small fraction of the bandwidth available on the LAN. The proliferation of the minicomputers was also of the motivations for the development and the deployment of X.25 based packet switch networks by the public network operators. From a protocol point of view, each vendor or even application had its own protocol suite and proprietary protocols were the rule. The TCP/IP protocol suite, which was proposed as an evolution of the ARPANET protocol suite could be considered as one of these proprietary protocols. However, it should be noted that the specifications and even reference implementations for the protocols of the TCP/IP protocol suite have always been freely available. This was not the case for protocol suites such as IBM's SNA or Novell's IPX/SPX.

The evolution of the data networks did not stop during the eighties. Instead, the introduction of the PCs and the workstations and later on their generalization have greatly

[1]This is one of the many acronyms that the readers of this thesis have to assimilate. To ease this process, we have included a list of acronyms in appendix A.

increased the need for local and wide area networks. This need was the driving force for the standardization of mature LAN technologies such as Ethernet or Token-Ring. These LANs were deployed in many companies during the eighties. Initially, these shared LANs were lightly loaded and several tens or even more computers were attached to a single shared LAN. When several disjoint shared LANs had to be interconnected, this was usually done by attaching them to a single "backbone" LAN (e.g. Ethernet or Token-Ring). This was possible because most of the traffic was local to these LANs. Measurements performed on these LANs showed that as a rule of thumb, 80% of the traffic was towards destinations attached to the same shared LAN as the source and only 20% of the traffic was towards destinations attached to a remote LAN. This rule is known as the 80/20 rule.

From a protocol point of view, the eighties were still dominated by proprietary and incompatible protocols. These incompatible protocols were the source of many problems when computers from different vendors had to be interconnected. In order to solve this problem, most vendors and several governments have supported the development of an open and non-proprietary protocol suite. The International Standardization Organization (ISO) spent a lot of time to develop a new non-proprietary protocol suite but we now know that this protocol suite did not achieve the expectations of most vendors and users. Instead, and partially due to the success of the Internet, the TCP/IP protocol suite became the dominant protocol suite and is currently replacing most of the proprietary protocol suites used during the eighties.

During the nineties, the "data networks" became an important part of the infrastructure of most organizations. Today, most, is not all, desktop computers are attached permanently (e.g. through a LAN) or semi-permanently (e.g. through a dialup modem) to a data network. While the PCs and workstations used in the eighties were rarely able to saturated the shared LANs, today even low-end PCs are fast enough to completely utilize an Ethernet LAN. Furthermore, with the proliferation of large software packages and graphic displays, the amount of traffic on LANs are steadily increasing. Due to this traffic increase, the number of workstations/PCs attached to each shared LAN has been continuously decreasing during the last decade. Today, ten workstations on a single shared LAN is a pretty large number, and we expect that soon each desktop computer will be attached to its "own" LAN. Besides, this, the LANs have also been improved to support higher speeds than their original speed (e.g. 100 Mbps and 1 Gbps Ethernet or 100 Mbps Token-Ring). Due to this evolution, the servers have moved from the shared LAN towards the backbone network and today the 80/20 rule of thumb is not anymore valid. In fact, some people suggest that today's rule of thumb should be that 20% of the traffic stays on the local LAN while 80% goes on the internetwork.

A similar increase in the available bandwidth happened in the wide area networks. Today, wide area links supporting several hundred megabits per second or a few gigabits per second are being actively deployed by network operator to sustain the growth of the traffic on the data networks. This evolution also affects the individual user since with the deployment of Digital Subscriber Line (DSL) or Hybrid-Fiber Coax (HFC) modem technologies each of these users will have a similar bandwidth than the Internet backbone ten years ago.

1.2 Beyond best-effort service

Most TCP/IP based networks and LANs such as Ethernet have been designed with a "fair best-effort" service in mind. "Fair" in this context means that if N users compete to utilize a single resource (e.g. bandwidth on a data network), then each user should receive $\frac{1}{N}$ of this resource. "Best-effort" in this context means that the network will do its "best" to provide the expected "fair" service, but without any precise commitment.

Such a best-effort service is suitable for some environment, but definitely not for all the possible environments. For example, in a corporate network, some applications (e.g. transaction processing, financial operations, electronic commerce, disk mirroring, ...) are much more important for the company than others (e.g. Internet access, email, print service, ...). In this case, the managers of the company could not accept that these unimportant applications consume resources inside the network at the detriment of the critical applications. In best-effort networks, a common solution to this problem is to increase the resources inside the network. Although this solution works since when there are too many resources the network is not congested and thus the critical applications do not have to compete for bandwidth with the other applications, it is not the most economical one, especially in the wide area where bandwidth is, and will continue to be, expensive.

A better solution would be allow the network manager to have a better control on the utilization of the network resources by all the applications used on the corporate network. Several solutions have been proposed to control the distribution of the network resources. In pure TCP/IP based networks, these solutions range from prioritizations mechanisms such as the utilization of the Type of Service (TOS) and "Precedence" fields in the Internet Protocol (IP) header [Alm92] [BBB$^+$98] to the more complex controlled-load and guaranteed services [WC97] introduced recently within the Internet community.

Instead of extending the TCP/IP protocol suite which was designed with the assumption of a "fair best-effort" service, an interesting alternative is to rely on the services provided by an underlying layer. In this respect, Asynchronous Transfer Mode (ATM) could definitely play a key role. ATM was specifically designed to support the high speed links required by today's TCP/IP networks in both the local and the wide area while providing a much more controllable service than the "fair best-effort" service of pure TCP/IP networks. Furthermore, ATM is currently being used by several Internet Service Provider (ISP) networks for its extensive traffic management capabilities.

This thesis is entirely focused on the utilization of ATM to support TCP/IP traffic. More precisely, we study ATM services which are able to guarantee a minimum bandwidth while still allowing the endsystems to transmit at a higher rate if the network is not congested. We consider that these services are a key building block for designing "controlled" TCP/IP networks. We study these ATM services from a quantitative point of view to evaluate whether they can efficiently support TCP/IP traffic.

1.3 Goals and scope of the thesis

To quantitatively evaluate the suitability of ATM to efficiently support TCP/IP services with a minimum guaranteed bandwidth, we rely on two widely used techniques : measurements and simulations.

The measurements performed with real systems are probably one of the best ways to quantitatively evaluate the performance of computer networks. These measurements are particularly interesting because all the factors which have an impact on the performance are present in the real systems. However, measurements have two important drawbacks. First, and obviously, they can only be used when the real system exists. This means that measurements cannot be used during the early design phases. The second drawback of measurements on real systems is that they are quite expensive to conduct. As a consequence, measurements are usually carried in small networks. We rely on measurements to evaluate the performance of TCP with the ATM services which are already supported by commercial products or networks.

Simulations are frequently used to study the performance of TCP in ATM networks. An interesting feature of simulations is that they can be used to evaluate the performance of services and protocols which are not already supported in real equipments. For this reason, simulations can be used to bridge the gap between analytical models and measurements. Compared with measurements, the main advantage of simulations is that they are less expensive. For this reason it is possible to study by simulations more complex networks than what can be achieved with measurements. Another advantage of simulations over measurements is that they can be used early in the design phase. The main drawback of simulations is that the simulation model is only an approximation of the real world. However, this approximation is usually close to the reality with a good simulation tool and always less coarse than in an analytical model. In this thesis, we rely on simulations to study the performance of TCP in ATM networks larger than our available equipment and also to evaluate the performance of ATM services which are not yet supported by available equipment.

For the simulations described in this thesis, we have used the STCP simulation tool [Man96] developed by Sam Manthorpe at Ecole Polytechnique Fédérale de Lausanne (EPFL) to study the behavior of TCP in ATM networks. The main characteristics of STCP and a brief description of our modifications to this simulation tool may be found in appendix B.

This thesis focuses on the integration of ATM under TCP/IP to provide services with a guaranteed minimum bandwidth. By integration, we mean that we utilize ATM as the underlying subnetwork technology to carry TCP/IP traffic. This thesis is structured as follows.

We first summarize in chapter 2 characteristics of the emerging ATM environment and discuss the possible alternatives to deploy ATM networks to support TCP/IP traffic. In chapter 3, we pursue our presentation of the emerging ATM environment by describing in details the different ATM services which have been defined by the standardization bodies to efficiently TCP/IP traffic in ATM networks. In the remainder of the thesis, we study the performance of TCP with the four main ATM services which are able to provide a guaranteed minimum bandwidth, namely Constant Bit Rate (CBR), Available Bit Rate (ABR), Variable Bit Rate (VBR) and Guaranteed Frame Rate (GFR). Throughout the thesis, we assume that the reader is familiar with the internal of the TCP protocol[2] .

[2]The reader who is not familiar with TCP will find a short overview of the data transfer mechanisms used by this protocol in appendix C.

In chapter 4, we present a detailed measurement study of the performance of TCP in a wide area ATM network which provides the CBR service. The main contribution of these measurements is that in such networks, the main factor which influences the performance of TCP is the conformance of the ATM traffic with the traffic contract.

In chapter 5, we study the conditions which must be met for a deployment of the ABR service in public networks. We show that in such networks, the complexity of the ABR service is a significant burden.

After having discussed the limitations of the CBR and ABR services in chapters 4 and 5, we then study in chapter 6 the performance of TCP with the VBR service. We first present measurements carried in a small ATM LAN which show the importance of the conformance of the ATM traffic with the traffic contract. We then rely on simulations to evaluate the performance of an ATM backbone network supporting the VBR service. Our measurements and simulations identify several problems with the VBR service.

We then focus our attention to the recently proposed GFR service which is expected to solve the problems of the VBR service. In chapter 7, we present a detailed simulation study of the performance of TCP with this GFR service. For these simulations, we consider the two example switch implementations which have been proposed within the ATM Forum. These simulations show that the GFR service category is able to better support TCP/IP traffic than the VBR service category, but we identify several problems with the two example switch implementations.

In chapter 8, we propose three solutions to solve the problems associated with the two example implementations. We first describe a shaping algorithm which significantly improves the performance of the two example switch implementations with one variant of the GFR service. We then propose an alternative switch implementation which is able to completely support the GFR service in FIFO-based ATM switches. Finally, we propose a modified scheduler which is better suited to support the GFR service than the current WFQ-like schedulers.

Finally, we present our conclusions and discuss some further work in chapter 9.

Chapter 2

The emerging ATM environment

2.1 Introduction

In this chapter, we briefly present the characteristics of the emerging ATM environment which are relevant for the remainder of this thesis[1].

Since ATM was chosen as the transfer mode for the Broadband Integrated Services Digital Network (B-ISDN), we first summarize the B-ISDN protocol reference model in section 2.2 and the relevant parts of its lower layers in section 2.3. We then briefly present the main functions of the control plane of the B-ISDN in section 2.4. Several protocols have been proposed to support TCP/IP in ATM networks; we summarize the main characteristics of these protocols in section 2.5. Finally, we discuss in section 2.6 our views on possible deployment scenarios for ATM in corporate and public networks.

2.2 The B-ISDN reference model

While the initial work on ATM began in the early eighties, ATM gained a lot of interest when it was selected as the transfer mode for the Broadband Integrated Services Digital Network (B-ISDN) in the late eighties. B-ISDN was considered by the International Telecommunications Union - Telecommunications sector (ITU-T) as the natural evolution of the Integrated Services Digital Network (ISDN). As such B-ISDN was expected to be mainly deployed by public network operators in wide area networks.

A major part of the initial set of ITU-T recommendations was the definition of a protocol reference model for the B-ISDN [ITU90] (figure 2.1). This protocol reference model evolved from the ISDN protocol reference model [ITU84] and includes several planes. The lowest layer of the B-ISDN Reference Model is occupied by the physical layer. In contrast with physical layers defined for the Open Systems Interconnection (OSI) Reference Model [ISO84], the Physical Layer of the B-ISDN reference model does not offer a bit-transmission service, but instead offers a service where the unit of transmission is 424 bits long. This size corresponds to the size of the standardized ATM cell. These ATM cells are the Protocol Data Units (PDUs) used by the ATM layer. The ATM Adaptation Layer (AAL) is used to enhance the service provided by the ATM layer ; for example one

[1]A more in-depth presentation of the ATM environment may be found in e.g. [Onv95] or [Pry93].

of the standardized AAL protocols provides the transfer of variable length Service Data Units (SDUs). The Control and User planes rely on the services provided by the physical, ATM and AAL layers.

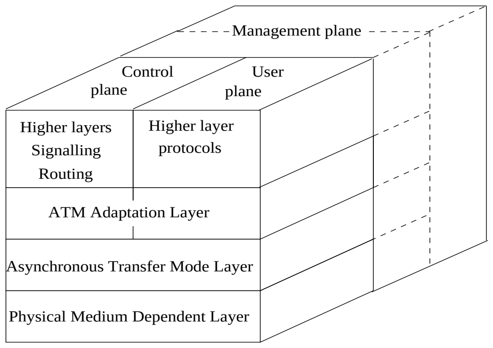

Figure 2.1: The B-ISDN Reference Model

The B-ISDN offers a connection-oriented service and the main function of the Control plane is to establish and release these connections. The User plane contains the functions which are used to perform the actual transfer of user information. The Management plane provides the management functions and is divided in two sub-planes. The Layer Management sub-plane contains the management functions which are specific to each layer of the B-ISDN reference model, while the Plane Management sub-plane performs the management and coordination functions related to the complete system.

In addition to the B-ISDN protocol reference model, the ITU-T has also defined reference interfaces which are an extension of the reference interfaces of the Narrowband Integrated Services Digital Network (N-ISDN). However, shortly after the publication of the first set of ITU-T recommendations on B-ISDN and ATM, several companies created the ATM Forum to speed up the standardization of ATM for corporate networks. The ATM Forum, while being only an industry consortium, has quickly become a key player in the standardization of ATM protocols and services. The work of the ATM Forum is as much as possible aligned with the work of ITU-T, but the ATM Forum is mainly concerned by the utilization of ATM in corporate networks, while ITU-T is mainly concerned by public networks. To structure its work, the ATM Forum has also defined a set of reference interfaces. As we will mainly refer to the ATM Forum specifications in this thesis, we will only discuss the interfaces defined by the ATM Forum.

The four main interfaces defined by the ATM Forum are shown in figure 2.2. The private User Network Interface (UNI) is the interface between an ATM user (e.g. a workstation with an ATM adapter) and an ATM network when both equipments are part of the same administrative domain (e.g. a corporate network). The public UNI is the interface between an ATM user (which may consist of a single workstation with an ATM adapter or a complete private ATM network) to a public service provider's ATM network. The Private Network-Network Interface (P-NNI) is the interface between two ATM switches which are part of the same private network. The Broadband Inter-Carrier Interface (B-ICI) the interface between two public ATM networks.

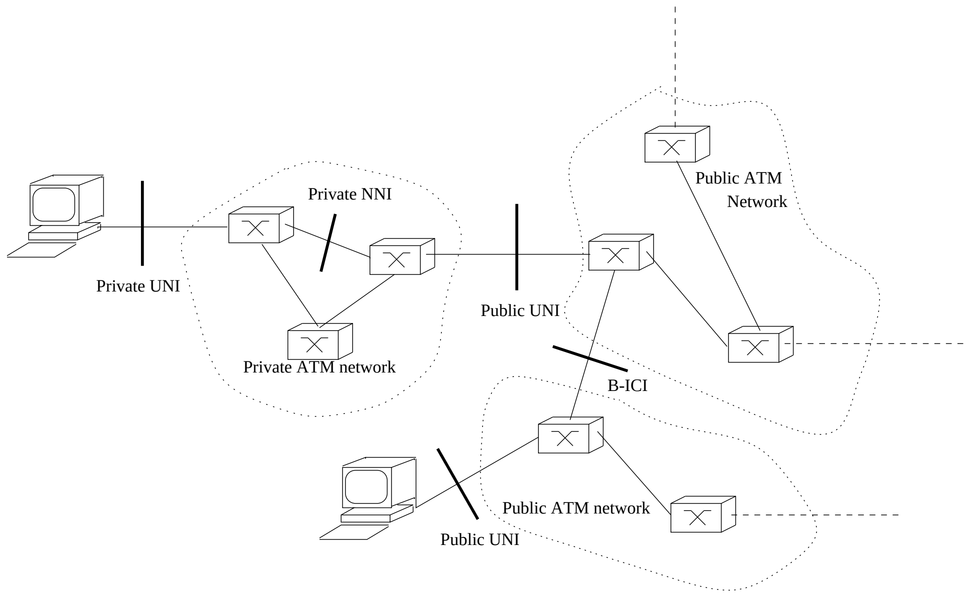

Figure 2.2: The reference interfaces defined by the ATM Forum

2.3 The lower layers of the B-ISDN Reference Model

The lower layers of the B-ISDN Reference Model provide services which are used by the User and the Control planes. In this section, we summarize the main functions and operations performed by the Physical, ATM and AAL layers.

2.3.1 The Physical layer

The Physical layer of the B-ISDN Reference Model is divided into two sublayers. The lower sublayer, i.e. the Physical Media Dependent sublayer (PMD), provides all the functions related to the physical transmission of bits. These functions include the electrical or optical transmission, the clock synchronization and the encoding functions. The upper sublayer (Transmission Convergence (TC)) provides all the functions related to the transmission and reception of ATM cells. These functions include the frame adaptation in the case of frame-based physical layer, the cell delineation and the generation and verification of the Header Error Check (HEC) (see section 2.3.2).

In this thesis, we will not consider the physical layer in details, but an important point to note is that there is not a single physical layer for the B-ISDN. The standardization bodies have already published specifications for a wide range of physical layers with throughput ranging from a few Mbps up to several hundreds of Mbps and most of these specifications are already implemented in commercial products. Furthermore, physical layers supporting ATM have been defined for Digital Subscriber Line (DSL) and Community Antenna Television (CATV) modems. Thus, private as well as public ATM network will be very often composed of different types of links, and this heterogeneity will be a major characteristic of ATM networks. In corporate networks, clients may be connected with 25 Mbps or 155 Mbps links, while 622 Mbps links and above may be used at the servers [Mic96] or for the backbone links. In public networks, link speeds of several gigabits per second or even more thanks to the utilization of Wavelength Division Multiplexing (WDM) will probably be used in the core of the network, while lower speeds will be used on the access links.

2.3.2 The ATM layer

The ATM (Asynchronous Transfer Mode) layer is the most important layer of the B-ISDN Reference Model. It contains all the functions related to the switching and the multiplexing of virtual connections. The unit of information in the ATM layer is a 53-bytes long ATM cell. Each ATM cell (figure 2.3) contains a 5-bytes long header, and 48 bytes of payload. The selection of a fixed unit of information transfer is one of the major decisions of ITU-T concerning B-ISDN.

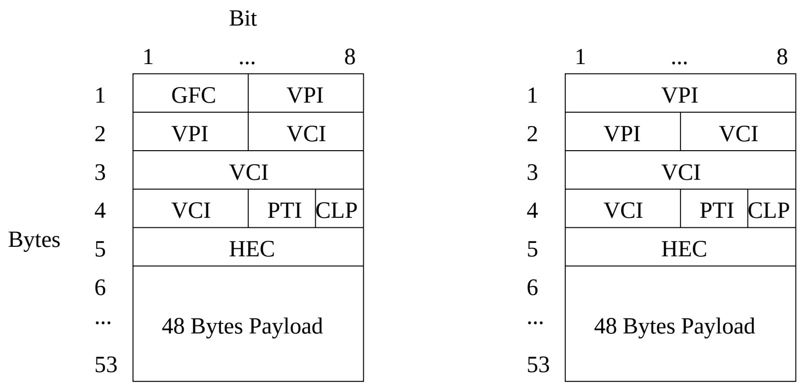

Figure 2.3: The ATM cell format at the UNI (left) and the NNI (right)

Two types of ATM cells have been specified. They differ in the usage of the first four bits of the header. At a UNI interface, these four bits are used to implement the Generic Flow Control (GFC) protocol. This protocol, standardized by the ITU-T [ITU93a], allows the network side of the UNI to regulate the flow of cells coming from the user side of the UNI. However, it should be noted that the amount of control allowed by this protocol is very limited, and thus it will probably only be used when simple devices are connected to an ATM network (especially at a public UNI). At a Network-Network Interface (NNI) interface, this protocol is not used, and the first four bits of the header are part of the Virtual Path Identification (VPI) field.

The ATM layer provides a connection-oriented service. Thus, an end-to-end (virtual) connection must be established before user information can be transmitted between ATM users. ATM supports two types of virtual connections. The permanent virtual channel connections (PVC) are established by configuration (e.g. at subscription time) or through management facilities. The switched virtual channel connections (SVC) are established and released dynamically by the signaling protocols of the control plane. In an ATM network, a route is associated with each virtual connection and all the cells transmitted for a particular virtual connection follow the same route. Furthermore, a major characteristic of a virtual connection is that the cell sequence is preserved (cell loss may occur, but cell reordering cannot occur). A two-level virtual connection hierarchy is used in ATM networks. The virtual channel connections (VCC) are usually end-to-end connections, while the virtual path connections (VPC) are usually established between two ATM switches to create virtual links inside the network or to aggregate a large number of VCCs in a single VPC to simplify the management of the network.

The Virtual Channel Identification (VCI) and VPI fields of the ATM cell header are used to identify the VCC and VPC to which the cell belongs. On a physical link, each end-to-end virtual connection is identified by the triplet $<$ *physical link*, *VCI*, *VPI* $>$, but this identification has only a local significance as each ATM switch may translate the VCI and VPI fields of each cell that it forwards. There are two types of ATM switches : the VP switches and the VP+VC switches. Inside an ATM switch, the routing of cells is usually performed according to a table. This table is updated each time a virtual connection is established or released. In a VP+VC switch, this table provides the mapping from the triplet $<$ *incoming physical link*, *incoming_VCI*, *incoming_VPI* $>$ to the triplet $<$ *outgoing physical link*, *outgoing_VCI*, *outgoing_VPI* $>$. When a cell is received on an incoming physical link, the table is consulted, and the cell is forwarded to the outgoing physical link indicated by the table, with incoming_VCI replaced by outgoing_VCI, and incoming_VPI replaced by outgoing_VPI. A VP switch is simpler since it does not look at and does not modify the VCI field when it forwards a cell.

The Payload Type Indicator (PTI) is used to distinguish between several types of ATM cells (table 2.1). The first bit of the PTI field indicates whether the cell carries some user data or whether it contains some network-related information. For user data cells, the second (Explicit Forward Congestion Indication (EFCI)) bit indicates whether the cell has experienced congestion in the network. The utilization of this EFCI bit will be discussed in section 3.5. The third PTI bit for user data cells is used by the ATM Adaptation Layer (see section 2.3.3). Three types of network-related cells are currently defined. The Operation and Maintenance (OAM) cells are used by the Management plane, notably, to detect loss of connectivity and to gather some performance measurements. The Resource Management (RM) cells are used to control congestion inside the network (see sections 3.5 and 3.7).

PTI	Meaning
000	User data cell, congestion not experienced (SDU type 0)
001	User data cell, congestion not experienced (SDU type 1)
010	User data cell, congestion experienced (SDU type 0)
011	User data cell, congestion experienced (SDU type 1)
100	Segment OAM flow-related cell
101	End-to-end OAM flow-related cell
110	Resource Management (RM) cell
111	Reserved

Table 2.1: Payload Type Indicator

The Cell Loss Priority (CLP) is a one-bit field that is used to distinguish between two cell loss priorities. A cell with CLP=0 is a high priority cell and a cell with CLP=1 is a low priority cell. It is expected that when the network is congested, the CLP=0 cells should be discarded after the CLP=1 cells.

The Header Error Check (HEC) is an 8-bit Cyclical Redundancy Check (CRC) that is used the detect bit errors in the cell header and also with some physical layers to correct single bit errors. The HEC is computed by each ATM entity (i.e. endsystem or switch) that sends a cell and verified by each ATM entity (i.e. endsystem or switch) that receives a cell. An ATM entity which receives a cell with an invalid HEC must discard this cell.

2.3.3 The ATM Adaptation Layer

The initial ITU-T recommendations published in 1990 expected that four major classes of applications would benefit from B-ISDN. These applications were classified in four classes according to their requested bit rate, the existence of a timing relation between the source and the destination, and whether they required a connectionless or connection-oriented service (table 2.2). It was also expected that each class of application would be served by one ATM Adaptation Layer (AAL).

	Class 1	Class 2	Class 3	Class 4
Bit rate	constant	variable	variable	variable
Timing relationship	yes	yes	no	no
Connection-mode	CO	CO	CO	CL

Table 2.2: The initial proposed service classes

The ITU-T began to define the AAL protocols and services for each service class. AAL1 [ITU91] was developed for applications in class 1 (e.g. uncompressed voice and video, leased line emulation). The AAL protocol for class 2 (AAL2) was recently defined. It shall be used by applications that transmit a variable amount of data with timing requirements (e.g. compressed voice). As the requirements for classes 3 and 4 were similar, the ITU-T decided to define a single AAL (AAL3/4) [ITU91] to cover the needs of applications from class 3 and class 4. However, due to its complexity combined with its large overhead, AAL3/4 did not gain much acceptance. To overcome the limitations of AAL3/4, ANSI and the ATM Forum proposed a much simpler AAL protocol [Lyo91], under the name AAL5. AAL5 has been designed to fulfill the needs of the applications in classes 3 and 4 with a much lower overhead than AAL3/4 and to permit a high-speed hardware implementation. It quickly became widely accepted, and today most applications corresponding to classes 3 and 4 rely on AAL5. We only consider AAL5 in this thesis.

AAL5 [ITU96a] provides the unassured transfer of variable-sized SDUs over the underlying ATM network. Thus, AAL5 hides the cell-based ATM network to the upper layer protocols and offers a classical packet-based service to those protocols. The AAL layer is itself divided into two sublayers. The Segmentation and Reassembly (SAR) sublayer contains all the functions related to the transmission and reception of the ATM cells corresponding to a variable-size SDU. The Convergence Sublayer (CS) contains all the functions related to the encapsulation of variable sized SDUs, and optional additional functions. It is also divided into two sublayers. The Common Part Convergence Sublayer (CPCS) provides the functions that are necessary for all users of AAL5. The optional Service Specific Convergence Sublayer (SSCS) contains the functions that may be used to provide an enhanced service (e.g. a reliable service) for some AAL5 users. For example, the UNI signaling protocol (see section 2.4.1) relies on an assured service provided by the Service Specific Connection Oriented Protocol (SSCOP) [ITU94a] in the SSCS sublayer over AAL5. By default, and in particular when used to carry IP traffic, AAL5 contains a null SSCS sublayer. We will thus only consider a null SSCS sublayer in this thesis.

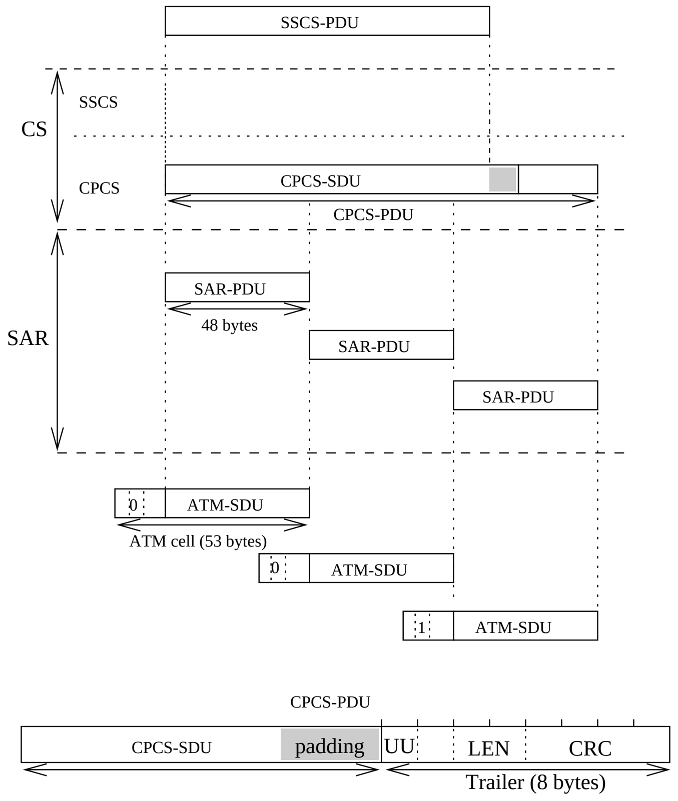

Figure 2.4: AAL5 segmentation

The AAL5 CPCS sublayer is responsible for the encapsulation of a variable sized-SDU submitted by the AAL5 user. AAL5 supports byte-aligned SDUs with sizes ranging from one byte up to 65535 bytes. The AAL5 CPCS encapsulation (figure 2.4) consists of some padding bytes and an 8-bytes long trailer. The padding is used to align the AAL5 CPCS-PDU on 48 bytes boundaries (i.e. a fixed number of cells). The CPCS-trailer contains a one-byte field (User to User indication (UU)) that is transmitted transparently between the AAL5 users together with the CPCS-SDU. The LEN field contains the length of the CPCS-SDU in bytes. This field is used by the receiver to remove the padding bytes, but also to verify that the received CPCS-PDU has a valid length. The last field of the CPCS-trailer is a 32 bits CRC that is used to verify that a received CPCS-PDU has not been corrupted during transmission.

The segmentation of a CPCS-PDU by the AAL5 SAR sublayer does not introduce any overhead in addition to the overhead of the CS sublayer. It relies on the PTI field of the ATM cell headers("SDU type 0" and "SDU type 1" in table 2.1). The utilization of this PTI field by the AAL5 SAR sublayer is another violation of the layering principle, but these violations are sometimes necessary to achieve a good performance. When an AAL5 CPCS-PDU is transmitted by the SAR sublayer, all its cells are transmitted as "SDU type 0" cells, except the last cell (which is transmitted as an "SDU type 1" cell). As the cell sequence is preserved on an ATM connection, the receiver can detect the end of a CPCS-PDU when it receives a "SDU type 1" cell. The advantage of this segmentation and reassembly mechanism is that it does not introduce any overhead, but its disadvantages are that it is impossible to multiplex cells from several CPCS-PDUs on a single VC, and that the loss of the last cell of a CPCS-PDU will usually cause the concatenation of the CPCS-PDU containing this last cell with the subsequent CPCS-PDU, and thus the loss of two CPCS-SDUs. Despite these limitations, AAL5 is probably today the most widely used AAL. It is used for legacy protocols (Classical IP over ATM, LAN Emulation), but also for protocols in the control plane (SSCOP, ILMI, ...) and for multimedia protocols such as the transmission of MPEG streams.

2.4 The Control plane

The Control plane contains all the functions which are necessary to establish and release switched virtual circuits and virtual paths. These functions include the signaling protocols for both the UNI and the NNI interfaces, and the routing protocol used by the ATM switches. As we are mainly concerned by the endsystems in this thesis, we will only briefly review the UNI signaling protocol. The ATM Forum NNI signaling and routing protocols are specified in [For96b], but will not be further discussed in this thesis.

2.4.1 UNI signaling

The UNI signaling protocol is built upon the AAL5 service enhanced with a SSCS sublayer containing the SSCOP [ITU94a] and the Service Specific Coordination Function (SSCF) [ITU94b]. SSCOP provides a reliable delivery of the UNI signaling messages. For this, SSCOP uses mechanisms such as flow control, error detection, retransmissions, etc. The SSCF is used to map the particular requirements of the UNI signaling layer to the lower layers.

The UNI signaling protocol is derived from the N-ISDN signaling protocol [ITU93b]. It relies on the reliable service provided by SSCF/SSCOP and allows an endsystem to establish point-to-point and point-to-multipoints virtual connections through an ATM network. For this, the UNI signaling protocol uses several types of variable-length messages. The UNI signaling protocol is only used at the UNI interfaces of the network ; the establishment of virtual connections inside the network is usually done by another protocol (e.g. the signalling part of P-NNI [For96b]). A typical establishment of a virtual connection is depicted in figure 2.5. If the "Source" endsystem wants to establish a virtual connection to the "Destination" endsystem, its signaling entity must first send a SETUP message through its UNI interface. The SETUP message is sent on a well-known VCI that connects the endsystem to the network-side of the UNI. The SETUP message may contain several parameters, but the main ones are the address of the "Destination" endsystem, the requested service category and the associated traffic contract (see chapter 3) and some information for the layers above the ATM layers.

The ATM Forum UNI signaling messages support two types of addresses. The first type of addresses are the E.164 addresses that are assigned from the ISDN numbering plan.The E.164 addresses will typically be used by ATM endsystems connected to public networks. Besides the E.164 addresses, the ATM Forum UNI signaling protocol also supports the “private ATM network addresses”. These addresses are modeled after the OSI NSAP format and are 20 octets long. The high-order part (octets 1-13) can be considered as the network part, while the low-order part (octets 14-20) can be considered as the endsystem part. The high-order part can be divided in subfields to create several levels of hierarchy and it may also contain an E.164 address [For93]. The 6 high-order octets of the endsystem part usually contain a unique endsystem identifier (e.g. IEEE 48 bits MAC address) while the low-order octet is a selector that may be used internally by the endsystem to distinguish different devices.

Unlike the telephone network, which was designed to support a single application, B-ISDN was designed from the beginning as a multi-service network. This means that B-ISDN must be able to support the requirements of a wide range of applications. When an application requests the establishment of a virtual connection, it shall specify some of its requirements as a requested service category and a traffic contract. Several service categories have been defined by the standardization bodies :

- Constant Bit Rate (CBR)
- Variable Bit Rate (VBR)
- Unspecified Bit Rate (UBR)
- Guaranteed Frame Rate (GFR)
- Available Bit Rate (ABR)
- ATM Block Transfer (ABT)
- Controlled Transfer (CT)

We will describe the different service categories and their associated traffic contracts in chapter 3. The service category indicates to the network the type of “service” that the user expects to receive on the established VC. Besides this information, the network also needs more information on the amount of traffic that the user will transmit on this VC. This information is contained in the traffic contract associated with the VC. A traffic contract is associated with each direction of transmission in case of bidirectional virtual connections, and the traffic descriptors used on the two directions may differ. The traffic descriptor is conveyed in the ATM User Cell Rate, Broadband Bearer Capability and QoS parameter information elements in the UNI 3.0 SETUP message.

Besides the remote address and the traffic descriptors, that are used by the ATM network when establishing a virtual circuit, the SETUP message may also contain some parameters that are transmitted transparently by the network. For example, the “AAL parameters” information element is used to indicate the type of AAL protocol that will be used on the VC (i.e. AAL1 or AAL3/4 or AAL5), but it is also used to negotiate some parameters of the AAL layer between the source and the destination endsystem. When

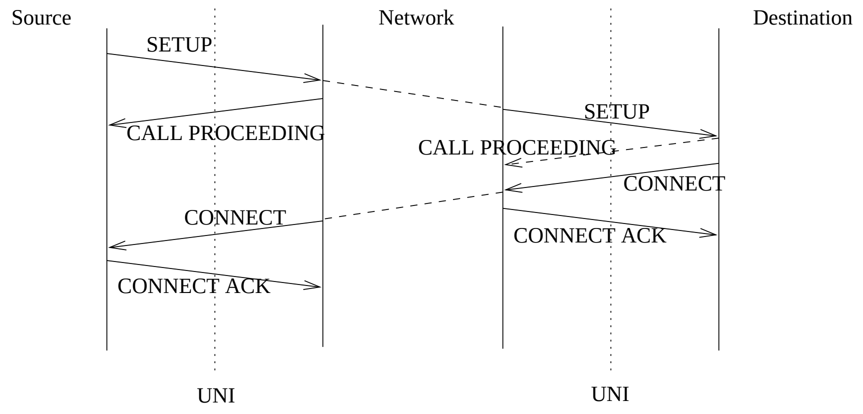

Figure 2.5: VC establishment with UNI signaling

AAL5 is selected, the source endsystem may also propose a maximum CPCS-SDU size for the forward and the backward VCs.

When the SETUP message is received by the ATM switch connected to the “Source”, it will acknowledge it by sending a CALL PROCEEDING message, and it will try to find a route towards the “destination” endsystem. If enough resources are available to establish the requested virtual circuit, the ATM switch will forward the VC establishment request with the NNI signaling protocol [For96b] to the ATM switch connected to the “destination” endsystem. If enough resources are available, this ATM switch will send a SETUP message to the “destination” endsystem. This SETUP message will contain the information found in the SETUP message sent by the “source” endsystem, as well as the VPI/VCI label to be used for this VC. The “destination” endsystem may acknowledge the SETUP message by sending a CALL PROCEEDING message. If the “destination” endsystem accepts the VC establishment, it will send a CONNECT message across its UNI. The “destination” ATM switch will forward the CONNECT message to the source ATM switch, and confirm the establishment of the VC with a CONNECT ACKNOWLEDGE message. The “source ATM” switch will send a CONNECT message to the “source” endsystem, and this endsystem will respond with a CONNECT ACKNOWLEDGE. At this time, the “source” endsystem may begin to transmit on the newly established VC.

2.4.2 Usage Parameter Control

When ATM was designed during the 1980s, one of the major design decisions was that congestion in the network should be avoided as much as possible. To avoid congestion or at least to lessen it, it was decided to rely on an open-loop control mechanism. As mentioned previously, when a user requests the establishment of a virtual connection, it must specify the required service category and a traffic contract. Based on this traffic contract, each ATM switch traversed by this virtual connection can verify whether it has enough available resources to support the new connection without risking congestion or jeopardizing other already established connections.

Once the virtual connection has been accepted and is established, the user is allowed to transmit at the rate specified by its traffic contract. If a misbehaving or malicious user was transmitting at a higher rate than its traffic contract, it could cause congestion inside the network and disturb other virtual connections. This is of course not acceptable, especially in public networks. To avoid this problem, the ITU-T and the ATM Forum specifications recommend the utilization of a Usage Parameter Control (UPC) to monitor the cell flow on VCs and VPs across the Public UNI.

The role of a UPC is twofold. First, the UPC must verify the validity of the VPI/VCI of the ATM cells sent across the UNI. Second it must monitor the traffic entering the network on established VCs/VPs and verify that the traffic contract is not violated. For this, the UPC compares the arriving cell flow with an ideal cell flow of a reference user with the same traffic contract. When a cell arrives at a UPC, it will be declared conforming if it could have been generated by the reference user according to the traffic contract of its VC. Otherwise, the arriving cell is declared non-conforming. If the cell is conforming, it will be accepted inside the network. If it is non-conforming, the UPC may, at the discretion of the network operator [ITU96c] :

- accept the non-conforming cell unmodified in the network
- change the CLP bit of the cell from 0 to 1, if the non-conforming cell was a CLP=0 cell
- discard the non-conforming cell

The second option allows the UPC to accept non-conforming cells as low priority cells inside the network. If the network is not congested, these low priority cells may reach the destination, but if the network becomes congested, these low priority cells will be the first cells to be discarded. In the case of SVCs, the network may also initiate the release of this SVC if the amount of non-conforming cells is excessive.

2.5 The higher layers

The User and Control planes contain a rich set of protocols which provide similar services as the layers 1-4 of the OSI reference model. Applications which rely only on these protocols are often called “native ATM” applications. However, up to now, few “native ATM” applications have been developed or deployed. Today, and probably for the near future, ATM is mainly used as a subnetwork technology which carries the traditional data communication protocols (e.g. TCP/IP, IPX, Appletalk, APPN, ...)

To efficiently support these data communication protocols, and TCP/IP in particular, over ATM-based networks, three groups of solutions have emerged within the Internet Engineering Task Force (IETF) and the ATM Forum. The solutions in the first group are Classical IP over ATM and LAN Emulation. These two solutions mainly address the issue of how to map an IP (resp. IEEE MAC) address onto the associated ATM Service Access Point (ATMSAP) address used in the ATM subnetwork. They both rely on servers to perform this mapping. The second group of solutions are the Next Hop Resolution Protocol (NHRP) and Multi-Protocol over ATM (MPOA). These two solutions can be considered as enhancements to Classical IP over ATM and LAN Emulation in the sense that they try to better benefit from the capabilities of the underlying ATM network.

More recently, several companies have proposed proprietary protocols to achieve a closer integration between IP and ATM. These proprietary solutions could be considered as the third group of solutions. The IETF is currently working on a new standard based on these proprietary protocols. We will briefly discuss the main characteristics of these protocols in the following sections.

2.5.1 Classical IP over ATM and LAN Emulation

Classical IP over ATM [Lau94] was defined by the IETF as a short term solution to carry TCP/IP traffic over ATM networks. This solution considers the ATM network as equivalent to a legacy LAN segment and is based on the concept of a Logical IP Subnet (LIS). A LIS has the following characteristics :

- All the members of the LIS belong to the same IP subnet
- All the members of the LIS are attached to the ATM network
- All the IP traffic between the members of the LIS is carried on ATM VCs, but all the traffic whose destination is outside the LIS must pass through a conventional IP router which is part of the LIS, even if the destination is attached to the same ATM network
- There is a mechanism to resolve the IP addresses into ATMSAPs

The requirement to use a conventional router for all the traffic whose destination is outside the LIS is the main limitation of Classical IP over ATM. The main contribution of Classical IP over ATM was the definition of a server-based mechanism to resolve the IP addresses into ATMSAPs. In legacy LANs such as Ethernet, the resolution of the IP addresses into the corresponding IEEE MAC addresses is performed by the Address Resolution Protocol (ARP) which relies on the broadcast facility provided by these LANs. As ATM does not provide such a broadcast facility, the classical ARP protocol could not be reused. The IETF choose to use a server (the ATMARP server) to provide the address resolution inside each LIS. Upon initialization, each LIS member establishes an ATM VC to the ATMARP server of its LIS and registers the mapping of its IP address into an ATMSAP. When a member of the LIS wishes to send IP packet to another member of the LIS it first verifies if it has already established one ATM VC to the destination endsystem. If so, the IP packet is transmitted on this VC. Otherwise, the ATMARP server is queried to retrieve the ATMSAP corresponding to the destination of the packet and a direct ATM VC is established once the ATMSAP address is known. This ATM VC is then used for all the traffic between the two endsystems. The current IETF specifications [PMH$^+$95] suggest the utilization of the UBR service category (i.e. best effort service) for these ATM VCs, but does not preclude the utilization of other service categories.

Besides the ATMARP server, Classical IP over ATM also recommends the utilization of the LLC/SNAP encapsulation to carry network layer traffic, including TCP/IP, over ATM [Hei93]. Another important point to mention is that the IETF has chosen a default Maximum Transmission Unit (MTU) size of 9188 bytes (i.e. 192 ATM cells) for IP over

ATM, although two endsystems may negotiate a different MTU via the ATM signalling protocol.

While the IETF has concentrated its work on the support of IP, the ATM Forum has chosen to define a service to emulate the service offered by the MAC layer in legacy LANs such as Ethernet or Token Ring. This service is called LAN Emulation.

With LAN Emulation [For95], the endsystems implementing the LAN Emulation protocols are grouped in Emulated LANs. Each Emulated LAN contains three distinct servers (the LAN Emulation (LANE) servers). These servers perform the functions required to emulate the MAC service provided by an Ethernet or a Token-Ring LAN. The three LANE servers are the LAN Emulation Configuration Server (LECS), LAN Emulation Server (LES) and Broadcast and Unknown Server (BUS). Upon initialization, each endsystem attached to an Emulated LAN establishes an ATM VC to the LECS of this Emulated LAN to retrieve the operational parameters of this Emulated LAN. The main parameters are the type of Emulated LAN (Ethernet or Token Ring) which defines the format of the frames sent in the Emulated LAN, the Maximum Data Frame size and the ATMSAP address of the LES. The Maximum Data Frame Size is used because LAN Emulation does not include a fragmentation mechanism. Thus all the endsystems attached to an Emulated LAN must utilize the same maximum frame size. Four different maximum data frame sizes are supported : 1516, 4544, 9234 and 18190 bytes. The BUS is used notably to emulate the broadcast service.

The LES provides the resolution of IEEE MAC addresses into ATMSAP address. After having contacted the LECS, an endsystem registers the mapping of its IEEE MAC address into its ATMSAP address at the LES. Thus, the role of the LES is similar to the role of the ATMARP server in Classical IP over ATM [2]. When a member of an Emulated LAN needs to communicate with another member of the Emulated LAN, it first retrieves the ATMSAP address of the other endsystem from the LES and establishes an ATM VC to this endsystem. With LAN Emulation 1.0, this VC must use the UBR service category. This limitation has been removed in LAN Emulation 2.0 [For97]. In addition to supporting the other service categories, LAN Emulation 2.0 also allows the LECS to distribute "suggested" service categories and traffic contracts to the members of the Emulated LAN.

2.5.2 MPOA and NHRP

LAN Emulation and Classical IP over ATM discussed in the previous sections are suitable for small networks, but they are not usable in large network. For larger networks, the IETF and the ATM Forum have proposed two similar solutions : NHRP and MPOA. These two solutions rely on the same basic idea, although NHRP is tailored for IP while MPOA aims at supporting other network layer protocols than IP (e.g. IPX and Appletalk). Compared to Classical IP over ATM and LAN Emulation, the main advantage of NHRP and MPOA is that they allow the endsystems to establish direct ATM VCs outside the boundaries of the LIS (resp. Emulated LAN). In this section, we only describe MPOA since NHRP is

[2] When IP is used in an Emulated LAN, the resolution of the IP addresses into the MAC address is done with the ARP protocol, while the resolution of this MAC address into an ATMSAP is performed by the LES.

a part of MPOA (the reader interested by the details of the complete IETF solution is referred to [LKP+98]).

MPOA can be considered as an extension of LAN Emulation. Inside the Emulated LAN, all the communications between the members of the LAN are done through the LAN Emulation protocol. When an endsystem wishes to communicate with an endsystem which resides in another Emulated LAN, it may use the MPOA services to directly establish an ATM VC to this endsystem. Thus with MPOA all the traffic between two Emulated LANs does not need to be forwarded by a bridge or a router as with LAN Emulation. With MPOA, each Emulated LAN is served by an MPOA Server (MPS). The role of this server is to map network layer addresses (e.g. IP address) in ATMSAPs. When an endsystem wants to send some frames to another endsystem outside the Emulated LAN, it requests its MPS to provide the resolution of the network layer address of the endsystem into an ATMSAP address. The ATMSAP address provided through the MPS may be either the ATMSAP address of the destination endsystem or the ATMSAP address of the egress router which is closest (from a network layer routing protocol point of view) to the destination endsystem. The resolution of the network layer address is either performed directly by the MPS or by querying other MPSs. Once the source endsystem knows the ATMSAP address which may be used to reach the destination endsystem, it can establish an ATM VC to this address and send its traffic on this VC.

Like LAN Emulation 2.0, MPOA also supports other service categories than UBR. When an endsystem requests the resolution of a network layer address to the MPS, it can indicate in the request the preferred service category and traffic contract for the traffic destined for this network layer address, and the ATMSAP returned by the MPS will be accompanied by a preferred service category and ATMSAP for the destination address. Thus, by configuring the endsystems and the MPS, a network manager can have some control on how the service categories and the associated traffic contracts are used by MPOA in his ATM network.

2.5.3 IP switching and related proposals

Classical IP over ATM, LAN Emulation, MPOA and NHRP have a common characteristic : they all rely on the ATM Forum or ITU-T signalling and routing protocols to establish the required ATM SVCs. These solutions have a common drawback when considering large ATM-based IP networks. The first drawback is that the ATM Forum signalling and routing protocols are rather complex and the current implementations can only establish a limited number of VCs per second [NBM+97]. Another drawback is that two different addressing schemes and routing protocols need to be maintained. To better support IP traffic, several vendors [NMLH97, KNME97, ACMM97, RDR+97, WC98a] have recently proposed to replace the ATM Forum signalling protocols by a much simpler signalling protocol [ADF+98] and to use the classical IP routing protocols instead of the P-NNI routing protocol. In addition, they also advocate to integrate the ATM switches in the IP network by providing basic IP forwarding capabilities in each ATM switch.

The various proposals differ notably in how and when new ATM VCs are established. Some proposals establish a new ATM VC after some amount of traffic has been forwarded to a particular direction, while other proposals establish the ATM VCs solely based on the information provided by the routing protocols. We will not discuss these details here since they are not relevant for this thesis. The important point to note is that while the first proposals [NLM96] were only considering best-effort ATM VCs, there has been recently a clear evolution towards ATM VCs with other service categories, and notably to ATM VCs which are able to provide a minimum guaranteed bandwidth [NEH$^+$98, VR98].

Many believe that one of the key benefits of using ATM to support IP traffic is that it allows the network operator to have a fine control on the utilization of the resources in his network. By carefully establishing ATM VCs in his network, the network operator can ensure that his network is well balanced while providing the required QoS guarantees. In this case, we expect that VCs with a minimum guaranteed bandwidth will be a key feature to efficiently support IP traffic in ATM networks with the above mentioned protocols.

2.6 Deployment scenarios

In the previous sections, we have presented the main mechanisms and protocols which compose the emerging ATM environment. Before studying in details the performance of TCP/IP in this environment in chapters 4-8, it is useful to briefly discuss how ATM networks are being and will probably be deployed in the next few years.

When considering ATM deployment scenarios, we have to distinguish two types of networks : the corporate and the public networks. By public network, we mean a metropolitan or wide-area network built by a public network operator and which provides a service to a large number of customers. These customers can be either individual users or corporate networks. By corporate network, we mean the internal network of a company which is completely managed by this company. When the corporate network is composed of several geographically distributed local or campus networks, these networks are typically interconnected by using a service provided by a public network. We will first discuss the corporate networks in section 2.6.1 and the public networks in section 2.6.2.

2.6.1 ATM deployment in corporate networks

There have been several driving forces for the deployment of ATM-based corporate networks. When the first commercial ATM switches and adapters appeared [BCS93], the main advantage of ATM over the deployed LAN technologies (i.e. Ethernet and Token Ring) was the high bandwidth supported by the ATM switches and adapters. With ATM adapters and switches supporting 100 Mbps and then 155 Mbps links, ATM was a promising alternative, from a pure bandwidth point of view, to Ethernet and Token Ring. Compared with Fiber Distributed Data Interface (FDDI), the main advantage of ATM was its switched nature which meant that ATM could support much larger networks, with higher bandwidth links than a single FDDI ring.

Since the introduction of the first generation of ATM LAN equipment, the situation has drastically changed. ATM is not anymore the only switched network technology. Today, switches are available for the main LAN protocols, and often at lower prices than ATM switches. On the other hand, the legacy LANs have been largely improved during the last few years. The 100 Mbps version of Ethernet (FastEthernet) is now being widely deployed and a standard for a Gigabit version of Ethernet has been recently ratified. A 100 Mbps version of Token Ring is also being defined by the IEEE. Compared with

ATM switches and adapters providing similar peak bandwidth, the FastEthernet switches and adapters are usually much cheaper. This is one of the reasons why FastEthernet is currently more often deployed than ATM. On the server side, 622 Mbps ATM adapters and switches are available for some time, but they now have to compete with Gigabit Ethernet switches and adapters. Thus, it is clear that the reason to deploy ATM in a corporate LAN is not (at least anymore) to provide higher peak bandwidth to the desktop systems and the servers.

A second driving force for the introduction of ATM in the corporate network are the rich traffic management capabilities provided by ATM compared to the legacy LANs. These capabilities allow the ATM networks to support the Quality of Service (QoS) parameters required by real-time applications such as video-conferencing. However, the network managers can also use these capabilities to better utilize their network resources, especially on the wide area links and in the backbone. For example, the network manager of a bank may ensure that critical business applications such as disk mirroring or transaction processing have always at least a minimum amount of resources in the network, even during periods of congestion, and that the other less critical applications can utilize the bandwidth left unused by the critical applications when these critical applications are not active.

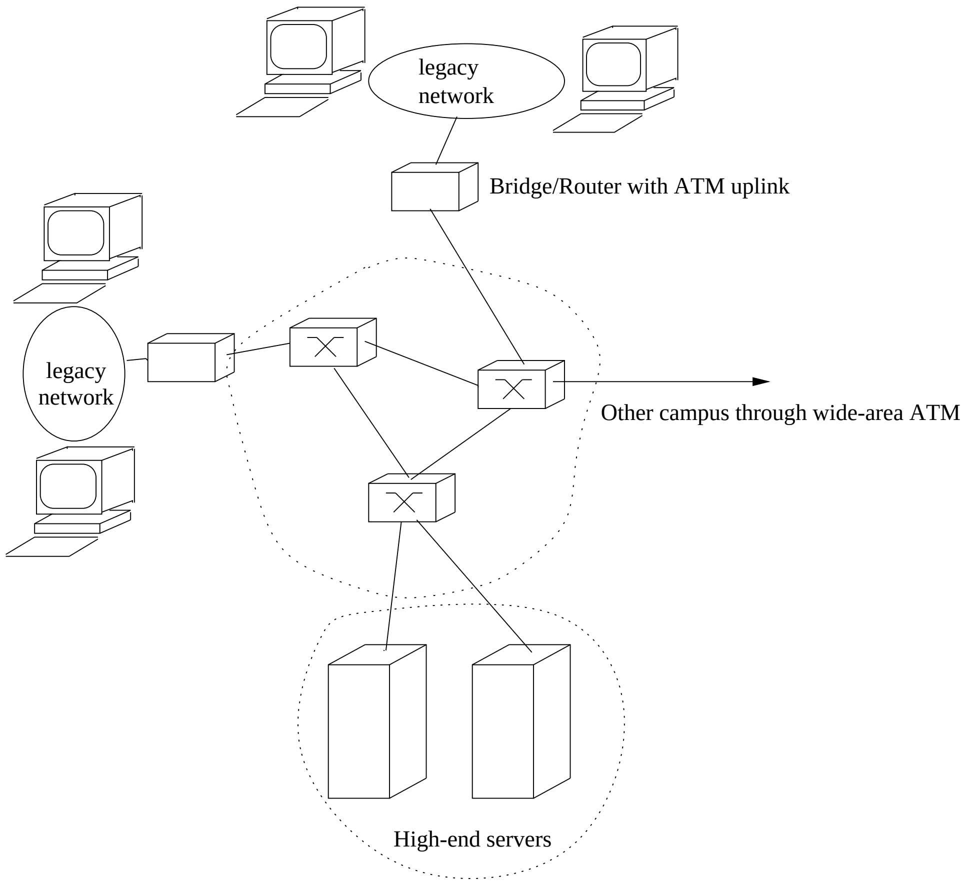
legacy network
Bridge/Router with ATM uplink
legacy network
Other campus through wide-area ATM
High-end servers

Figure 2.6: A typical ATM-based corporate network

Based on the evolution of the networking technologies during the last few years, a typical corporate ATM-based network may look as in figure 2.6. In such a corporate network, ATM is mainly used in the backbone. Most of the desktop endsystems are connected to legacy networks such as Ethernet or fast Ethernet. These legacy networks are attached to bridges or routers connected through ATM uplinks to the ATM backbone. A small number of ATM endsystems are directly attached to the ATM backbone. These endsystems are either high-end servers or server farms and desktop systems using specific applications such as video-conferencing. If the corporate network is composed of several geographically dispersed sites, the various subnetworks are interconnected with direct ATM links or ATM VPs from a public ATM network.

2.6.2 ATM deployment in public networks

In public networks, the driving forces for the deployment of ATM are different in the access and the core network. In the core network, the first and foremost driving force for the deployment of ATM is the fact that it supports different services. In the past, public network operators have traditionally deployed one separate network for each service (e.g. the telephone network, the Telex network, the X.25 network, the frame relay network,...) with few or no interconnection between these networks. Thanks to the traffic management capabilities of ATM, the public network operators will be able to federate all these different networks in a single multi-service ATM backbone while providing the same level of service as if they were still supported by different networks. This is currently the main reason to deploy ATM in the core of public networks. More recently, another driving force for the deployment of ATM has been the requirement to support higher bandwidths.

In the access network, the main driving force for the deployment of ATM is the need to provide which much higher bandwidth requirements than in the past. In the access network, most of the recent technologies providing bandwidth of 1 Mbps or above such as Digital Subscriber Line (DSL) or Hybrid-Fiber Coax (HFC) are used to transport ATM cells. Since ATM is used in the access network, this is another reason to also use this technology in the backbone.

When looking at TCP/IP, it can be expected that two major types of utilizations of the ATM networks will be important in public networks : Virtual Private Networks (VPNs) (e.g. used for LAN interconnection) and Internet access (for corporate and individual users). Today, LANs are usually interconnected either via leased lines or via data networks such as frame-relay. Compared with the leased lines, the main advantage of ATM is its flexibility. The main advantage of a leased line is that it provides a transparent connection between two sites at a fixed bandwidth. However, the cost of a leased line is often high, especially for high bandwidth lines and furthermore any modification to the leased line (e.g. bandwidth increase) requires the installation of a new physical circuit, which may take several months. ATM is much more flexible. ATM can provide a service equivalent to the leased lines with the CBR service category. If an ATM VC is used instead of a leased line then the bandwidth of the VC can be easily changed when needed, either through management for PVCs or through signalling for SVCs. Furthermore, a change in the characteristics of the ATM VC only requires a small re-configuration of the devices used to connect the LANs to the public network. But ATM can also provide a better LAN interconnection service with VCs with a minimum guaranteed bandwidth. The advantage of this solution is that when there is no congestion inside the network, each VC may utilize a higher bandwidth than its minimum guaranteed bandwidth. The efficient utilization of such VCs to carry TCP/IP traffic is the main subject of this thesis.

During the last years, the number of users connected to the Internet has increased rapidly. Most of the individual Internet users currently use analog modems or ISDN lines. While widely deployed, analog modems and ISDN lines provide a relatively low bandwidth. Internet access, as well as telecommuting would greatly benefit from higher bandwidth. It can be expected that, in the short term, many public network operators will deploy ATM-based ADSL modems to provide a service with a much higher bandwidth than ISDN for Internet access and telecommuters. We expect that the network operators will want to provide different types of services (with different tariffs) for different users. To work efficiently, a telecommuter will require some minimum guaranteed bandwidth during the office hours. On the other hand, a casual Internet user will probably be very happy with a low price best-effort service. In order to achieve a good utilization of its links, the network operator would expect that the casual Internet user could benefit from the bandwidth which is left unused by the telecommuter. However, the telecommuter would also like to be able to benefit form a higher bandwidth when the network is lightly loaded. An interesting way to support these two types of users is to provide a service with different minimum guaranteed bandwidths but which allows the endusers to benefit from unused bandwidth inside the network.

2.7 Conclusion

In this chapter, we have first summarized the main functions and protocols used to support TCP/IP in an ATM environment. We have then analyzed the possible deployment scenarios for ATM in corporate and public networks. In both cases, we have shown that the key benefits of ATM were its traffic management capabilities. We will discuss these capabilities in more details in chapter 3. We have also shown that VCs with a minimum guaranteed bandwidth were a key building block for corporate and public networks.

Chapter 3

The service categories

3.1 The evolution of the service categories

As explained in chapter 2, the initial work on B-ISDN was performed within the ITU-T in the late eighties. From the beginning, B-ISDN was designed as a network to support a wide range of possible application with various requirements. To support all these applications, it was clear that the B-ISDN would have to be aware of, at least, some of the requirements of each application. To achieve this, the ITU-T proposed to categorize all the possible applications according to three criteria :

- the required bandwidth (constant or variable)
- the timing requirements (real-time or non-real-time)
- the connection-mode (connection-oriented or connectionless)

As explained in chapter 2, four service classes and the corresponding ATM Adaptation Layers (AAL) have been defined based on the relevant values of these criteria (table 2.2).

In addition to the definition of one AAL for each service class, it was also necessary to define how the network should behave to support the QoS requirement from each service class. The first ITU-T recommendation [ITU93c] addressing this problem defined a set of general procedures for traffic and congestion control but did not entirely support the four service classes. This recommendation only supported VPs and VCs requiring a constant bandwidth. This constituted the basis for what is now known as the Constant Bit Rate (CBR) service category and will be discussed in details in section 3.2.

In November 1991, the creation of the ATM Forum by four computer companies has accelerated the development of ATM and of the other service classes in particular. The second major document produced by the ATM Forum [For93] defined not only the support for applications with constant bandwidth requirements, but also the support of applications with variable bandwidth requirement and also a new best effort service which was expected to be similar to the service offered today on the Internet. These two definitions are the basis for the Variable Bit Rate (VBR) and Unspecified Bit Rate (UBR) service categories that will be discussed in sections 3.3 and 3.4. The ITU-T has later incorporated the support for the VBR service category [ITU96c], but there are still ongoing discussions [SG198] on whether the ITU-T should explicitly support the best-effort service defined in [For93] or not.

While the UBR service category was, and still is, widely used in ATM LANs, the members of the ATM Forum considered that it was not a sufficient solution to provide a best-effort service in the long term. Shortly after the release of [For93], the ATM Forum started to work on a better than best-effort service under the name Available Bit Rate (ABR). The ATM Forum spent a considerable amount of time to specify this new service category whose definition appeared in [For96c] and that we will describe in details in section 3.5.

In parallel to the work on ABR, the ATM Forum has continued to work on the various service classes. The ATM Forum did not consider the "service class" model adopted in the early ITU-T recommendations (table 2.2) as the appropriate way to define the main requirements of the various applications that may benefit from the ATM networks. Instead of this model, the ATM Forum chose to adopt a new service model proposed in [Gar96]. This new service model first notes that the distinction between a connection-oriented and a connectionless service is not really appropriate in pure ATM networks since ATM is inherently connection-oriented. Then, it notes that the main distinction between all the applications is whether they require a real-time treatment or not. For the real-time applications, the ATM Forum has considered that their QoS requirements can be expressed with three independent parameters :

- Peak-to-peak Cell Delay Variation (ppCDV)
- Maximum Cell Transfer Delay (maxCTD)
- Cell Loss Ratio (CLR)

The CLR is defined for a connection as $CLR = \frac{Lost\ Cells}{Total\ Transmitted\ Cells}$. The maxCTD is the $(1-\alpha)$ quantile of the cell transfer delay for the cells on one connection and the ppCDV is the $(1-\alpha)$ quantile of the difference between the fixed cell transmission delay and the maxCTD. We will not consider these QoS parameters in this thesis since we are only concerned with non-real-time traffic.

For the real-time applications, the network shall not violate the QoS commitments agreed at the establishment of the VC. To accommodate the applications with fixed and variable bandwidth requirements, the ATM Forum has defined two (real-time) service categories :

- Constant Bit Rate (CBR)
- realtime Variable Bit Rate (rt-VBR)

However, most applications do not have the same timing requirements as the real-time applications. These applications can benefit from the three non-real-time service categories defined in [For96c] :

- Unspecified Bit Rate (UBR)
- non realtime Variable Bit Rate (nrt-VBR)

- Available Bit Rate (ABR)

The ITU-T did not adopt the same model as the ATM Forum and, up to now at least, ITU-T has been very reluctant to allow the endsystems to define precise QoS requirements for the real-time applications for each VC. Instead, the ITU-T chose to define a small set of QoS classes [ITU96b] where the values of several QoS parameters are specified and from which the endsystem can choose. In addition to the CBR, VBR and ABR service categories, the ITU-T has also defined a new service category called ATM Block Transfer (ABT) which is suitable for both real-time and non-real-time applications. We will describe it briefly in section 3.7.1.

Recently, the ATM Forum and the ITU-T have started to work on a new service category called Guaranteed Frame Rate (GFR). This new service category is suitable for the non-real-time applications. Its main characteristic is that it explicitly takes into account the fact that most endsystems transmit AAL5-PDUs. It can be expected that this new service category will become very important to support TCP/IP traffic in the near future. We will describe it in details in section 3.6. In addition to GFR, the ITU-T (but not the ATM Forum) has also recently been working on another service category called Controlled Transfer (CT). This service category is only suitable for non-real-time applications. We will describe it briefly in section 3.7.2.

3.2 The CBR service category

The Constant Bit Rate (CBR) service category is the simplest service category. Intuitively, the CBR service category aims at offering a service similar to the service offered by the Narrowband Integrated Services Digital Network (N-ISDN). In a N-ISDN network, calls are established with a bandwidth of $n \times$ *64 kbps* (usually $1 \leq n \leq 30$). This bandwidth granularity corresponds to the bandwidth required for a 4 kHz digital voice channel with 8 bits per sample. Inside a N-ISDN network, these 64 kbps channels are usually multiplexed on higher bandwidth links with the Time Division Multiplexing (TDM) technique.

In a network such as N-ISDN, relying on TDM multiplexing, the bandwidth of a channel can be expressed as the number of bits transmitted per second, or equivalently as the number of bytes allocated to this channel per 125 μsec slot. In an ATM network, the unit of transmission is a 53 bytes long cell instead of a single byte, but similar expressions can be used to define the bandwidth of a virtual circuit. The bandwidth allocated to a virtual circuit can equivalently be expressed as a cell rate (i.e. the number of cells transmitted during a period of one second), or as the interval between successive cells. The "cells per second" expression is ambiguous, and thus cannot be used to define a precise traffic contract used to determine the amount of resources required by the connection inside the network. For example, a virtual circuit that sends 100 cells, then becomes idle for almost one second will have the same "cells per second" rate as a virtual circuit which sends exactly one cell every 10 milliseconds. These two virtual circuits do not require the same resources (e.g. buffers) inside the network. To avoid this ambiguity, the ITU-T and the ATM Forum have chosen to specify the rate of a connection as the inter-arrival time between successive cells from the same connection. This inter-arrival time (or peak emission interval) is used by the Generic Cell Rate Algorithm (GCRA) chosen by the

ATM Forum and the ITU-T to unambiguously define a CBR traffic contract. The GCRA uses two parameters :

- T : the peak emission interval
- τ : the Cell Delay Variation Tolerance (CDVT)

T is the theoretical delay between successive cells and τ is a bound on the allowed variation of a cell flow from the theoretical arrival pattern. Two equivalent versions of the GCRA are proposed in [For93] and [ITU93c]. Throughout this thesis, we will refer to the Virtual Scheduling version of the GCRA, which can be expressed as shown in figure 3.1. The peak emission interval used in the GCRA is derived from the Peak Cell Rate (PCR) of the virtual connection expressed in cells per second as $T_{PCR} = \frac{1}{PCR}$. The CDVT depends on the characteristics of the physical layer used at the UNI, but also on the expected amount of multiplexing at the UNI.

```
Cell Arrival at time $t_a$ :


if( $t_a < TAT - \tau$ )
{
    /* non-conforming cell */
}
else
{
    /* conforming cell */
    $TAT = max(t_a, TAT) + T$;
}

$TAT$ : Theoretical Arrival Time (for Peak Cell Rate)
```

Figure 3.1: Pseudo-code for the Generic Cell Rate Algorithm

Since the first release of the ITU-T [ITU93c] and ATM Forum [For93] specifications, the CBR service category has been slightly modified. In the ATM Forum UNI 3.x specifications [For93] [For94a], the concept of service categories was not present, but three different CBR traffic contracts were defined at that time (table 3.1).

The first traffic contract does not distinguish between CLP=0 and CLP=1 cells. However, it can be expected that only CLP=0 cells will be sent when this traffic contract is used as charging is usually based on the PCR of the contract and CLP=1 cells have a higher probability of loss in case of congestion inside the network. The second and third traffic contracts define a PCR for the CLP=0+1 cell flow and a PCR for the CLP=0 cell flow. In these two traffic contracts, $PCR_0 \leq PCR_{0+1}$, and PCR_0 correspond to the fraction of the total traffic that carries CLP=0 cells. These two traffic contracts allow the endsystems to transmit both CLP=0 and CLP=1 cells. The only difference between the second and the third traffic contracts is that with the second traffic contract, the UPC may tag the non-conforming CLP=0 cells.

The first traffic contract of table 3.1 can be enforced by the GCRA shown in figure 3.1. The second and third traffic contracts of table 3.1, can be enforced by two GCRAs op-

Table 3.1: The three CBR traffic contracts in UNI 3.x

Peak Cell Rate for the CLP=0+1 cell flow (PCR_{0+1}) Cell Delay Variation Tolerance for PCR_{0+1} (τ_{0+1})
Peak Cell Rate for the CLP=0 cell flow (PCR_0) Cell Delay Variation Tolerance for PCR_0 (τ_0) Peak Cell Rate for the CLP=0+1 cell flow (PCR_{0+1}) Cell Delay Variation Tolerance for PCR_{0+1}(τ_{0+1}) Tagging of the non-conforming cells is requested
Peak Cell Rate for the CLP=0 cell flow (PCR_0) Cell Delay Variation Tolerance for PCR_0 (τ_0) Peak Cell Rate for the CLP=0+1 cell flow (PCR_{0+1}) Cell Delay Variation Tolerance for PCR_{0+1} (τ_{0+1}) Tagging of the non-conforming cells is not requested

erating in parallel (one on the CLP=0 cell flow and the other one on the CLP=0+1 cell flow).

While the CBR traffic contracts contain a cell delay variation tolerance parameter associated to each Peak Cell Rate, the UNI signaling protocols [For93, For96a] do not contain any information element to convey its value. In a real network, the CDV tolerance is usually chosen by the network operator as a function of the Peak Cell Rate (see for example the function used by the Swiss ATM Pilot cited in [JBLC96]). As the CDV tolerance is a margin allowed by the network on the traffic sent by the endsystem across the UNI, it seems natural to prohibit the endsystem from requesting a particular CDV tolerance for each VC. However, the enduser is usually not informed about the CDV tolerance chosen by the network operator and the UNI signaling protocols are not able to convey this information back to the endsystems.

The latest ATM Forum specifications on traffic management [For96c] and signaling [For96a] have slightly changed the definition of the CBR service category. The main modification is that the CBR traffic contracts involving PCR_{0+1} and PCR_0 defined in ATM Forum UNI 3.x are not supported by the latest Traffic Management specification. This specification only supports one type of CBR traffic contract (named CBR.1 - table 3.2). The two other CBR traffic contracts that were supported by UNI 3.x are not anymore supported by the latest Traffic Management specification [For96c], and the UNI 4.0 signaling protocol [For96a] only supports them for backward compatibility reasons.

Table 3.2: The CBR.1 traffic contract

Peak Cell Rate for the CLP=0+1 cell flow (PCR_{0+1}) Cell Delay Variation Tolerance for PCR_{0+1} (τ_{0+1})

While the first release of ITU-T recommendation I.371 [ITU93c] was aligned with ATM Forum UNI 3.x, there are several differences between the latest release of ITU-T I.371

[ITU96c] and the latest ATM Forum specifications concerning the CBR service category [1].

Like [For96c], [ITU96c] does not support the CBR traffic contracts where both PCR_{0+1} and PCR_0 are specified. [ITU96c] still supports the traffic contract where only PCR_{0+1} is specified, but it supports an additional traffic contract. This new traffic contract specifies a PCR and its associated CDV tolerance for the CLP=0+1 user cell flow and a PCR and its associated CDV tolerance for the OAM cell flow. This kind of traffic contract is not supported by the ATM Forum specifications.

3.3 The VBR service category

The CBR service category, discussed in the previous section is mainly suitable for applications requiring a constant amount of bandwidth for the whole duration of the virtual circuit. These applications include leased line emulation or uncompressed voice or video over ATM [For96c]. However, many applications do not have such strong bandwidth requirements. For example, the instantaneous bandwidth requirement of applications such as compressed voice or video usually varies during the connection from low values when few signal is transmitted (e.g. voice with silence suppression) to high values during short periods of time (e.g. during a scene change with compressed video) although the long term average bandwidth is much smaller than the peak bandwidth. Furthermore, most data applications (e.g. world wide web, NFS, ftp, email, ...) behave as ON-OFF sources. During the ON period, they usually try to utilize as much bandwidth as available, but they may be idle during long periods of time.

Among these applications, critical applications such as transaction processing (e.g. financial or banking transactions) or monitoring of real-time processes may desire to have some minimum amount of reserved bandwidth so that they always transmit a minimum amount of data, even when the network is heavily loaded. While these applications could rely on the CBR service category, this would produce a large waste of resources inside the network. With the latest CBR traffic contracts, this would also mean that those applications would not be able to transmit at a higher rate than the negotiated PCR_{0+1} even when the network is lightly loaded.

To fulfill the needs of these applications, the ATM Forum has introduced the Variable Bit Rate (VBR) service category in the UNI 3.0 specification [For93]. Intuitively, the VBR service category allows an endsystem to reserve some average amount of bandwidth inside the network, while still being allowed to transmit at a higher rate during short periods of time. More precisely, the VBR traffic contract is composed of five parameters :

- Peak Cell Rate (PCR)
- Cell Delay Variation of the Peak Cell Rate (τ_{PCR})
- Sustainable Cell Rate (SCR)

[1] There are several differences between the terminologies used by the ATM Forum and the ITU-T. For example, CBR is called DBR (Deterministic Bit Rate) by the ITU-T recommendations. To avoid changes of terminology throughout the thesis, we have chosen to always use the ATM Forum terminology even when describing mechanisms supported only by the ITU-T.

- Cell Delay Variation of the Sustainable Cell Rate (τ_{SCR})
- Maximum Burst Size (MBS)

The PCR and its associated CDV tolerance have the same meaning as with the CBR service category. The SCR corresponds intuitively to a long-term average cell rate and the MBS corresponds to the largest burst of cells than can be emitted at the PCR.

Formally, a VBR traffic contract is defined by two GCRAs. The first one (GCRA(T=1/PCR,$\tau = \tau_{PCR}$)) specifies the conformance with the PCR part of the traffic contract. The conformance to the SCR and MBS parts of the traffic contract are also defined by a GCRA. This GCRA uses an emission interval of 1/SCR, and a CDV tolerance (τ_S) defined as [For93] :

$$\tau_S = \tau_{SCR} + (MBS - 1) \times (\frac{1}{SCR} - \frac{1}{PCR}) \tag{3.1}$$

It can be shown by an inspection of figure 3.2 that with parameters computed as in equation 3.1, GCRA(T=1/SCR,$\tau = \tau_S$) will accept a burst of MBS cells sent at the PCR as conformant with the traffic contract. It can also be shown [For93] that over an interval of length t, the number of cells emitted with a spacing of at least T and still in conformance with the second GCRA is bounded by :

$$N(t) \leq min(\lfloor 1 + \frac{t + \tau_S}{T_{SCR}} \rfloor, \lfloor 1 + \frac{t}{T} \rfloor) \tag{3.2}$$

The first part of equation 3.2 applies for long periods of time ($t \geq MBS \times T$) while the second part of this equation applies for short periods of time. These two GCRAs are associated as shown in figure 3.2.

The ATM Forum UNI 3.x specification [For93] and the latest ITU-T I.371 recommendation [ITU96c] support three different VBR conformance definitions (table 3.3). These three conformance definitions differ in their respective support for CLP=0 and CLP=1 cells. With the VBR.1 conformance definition, all the parameters of the traffic contract are applicable to the CLP=0 and CLP=1 cells. In this case, the endsystem can send cells at PCR, but only if the burstiness of its cell flow is bounded by GCRA($1/SCR, \tau_S$). With the VBR.2 conformance definition the CLP=0 cell flow is bounded by GCRA($1/SCR, \tau_S$) but the endsystem is allowed to send CLP=1 cells in excess (provided that the CLP=0+1 cell flow is bounded by GCRA($1/PCR, \tau_{PCR}$)). With the VBR.3 conformance definition, the endsystem is allowed to send CLP=0 cells at PCR and the UPC at the ingress of the network will tag the cells which are declared non-conformant by GCRA($1/SCR$,τ_S). With the VBR.2 and VBR.3 conformance definitions, the CLP=1 cells will be carried on a best-effort basis inside the network.

The main advantage of the VBR.2 and VBR.3 conformance definitions over the VBR.1 conformance definition is that they allow the endsystem to have some minimum guaranteed bandwidth (defined by GCRA($1/SCR, \tau_S$)) while still being able to benefit from the available bandwidth inside the network on a best-effort basis.

```
Cell Arrival at time $t_a$ :


if( $t_a < TAT_{PCR} - \tau_{PCR}$ )
{
    /* non-conforming cell for PCR */
}
else
{
    /* conforming cell for PCR */
    $TAT_{PCR} = max(t_a, TAT_{PCR}) + T_{PCR}$;
    if( $t_a < TAT_{SCR} - \tau_S$ )
        {
            /* non-conforming cell for SCR or MBS */
        }
    else
        {
            /* conforming cell for SCR and MBS */
            $TAT_{SCR} = max(t_a, TAT_{SCR}) + T_{SCR}$;
        }
}
```

Figure 3.2: Pseudo-code for the VBR GCRA

Table 3.3: The three VBR conformance definitions in UNI 3.x

VBR.1	Peak Cell Rate for CLP=0+1 cell flow (PCR_{0+1}) Cell Delay Variation Tolerance for PCR_{0+1} ($\tau_{PCR_{0+1}}$) Sustainable Cell Rate for CLP=0+1 cell flow (SCR_{0+1}) Maximum Burst Size for CLP=0+1 cell flow (MBS_{0+1})
VBR.2	Peak Cell Rate for CLP=0+1 cell flow (PCR_{0+1}) Cell Delay Variation Tolerance for PCR_{0+1} ($\tau_{PCR_{0+1}}$) Sustainable Cell Rate for CLP=0 cell flow (SCR_0) Maximum Burst Size for CLP=0 cell flow (MBS_0) Tagging of the non-conforming cells by the UPC is not requested
VBR.3	Peak Cell Rate for CLP=0+1 cell flow (PCR_{0+1}) Cell Delay Variation Tolerance for PCR_{0+1} ($\tau_{PCR_{0+1}}$) Sustainable Cell Rate for CLP=0 cell flow (SCR_0) Maximum Burst Size for CLP=0 cell flow (MBS_0) Tagging of the non-conforming cells by the UPC is requested

3.4 The UBR service category

The Unspecified Bit Rate (UBR) service category tries to provide a service equivalent to the best-effort service of today's Internet in ATM networks. As such, the UBR service category is mainly suited for data applications built above protocols such as TCP/IP.

In fact, most of today's ATM LANs rely exclusively on this service category. The main advantage of the UBR service category over the other service categories is that it does not require any particular behaviour from the endsystems. With the UBR service category, the endsystems usually do not have to take special precautions to fulfill a precise traffic contract unlike with the CBR and VBR service categories. From an implementation point of view, the UBR service category is the easiest service category to support in the endsystems.

It should be noted that, like the best-effort IP service, the UBR service category does not provide any guarantee at all. This means that when an endsystem transmits some cells on a UBR VC, it can only hope that these cells will eventually reach their destination. This is due to the fact that when a switch providing the UBR service category is congested, the only mechanism it can use is to discard cells. The UBR service category does not define any access control or congestion control mechanism, and it explicitly assumes that if congestion occurs inside the network, congestion collapse will be avoided by the upper layer protocols such as TCP. Up to now, the UBR service category has been mainly used in ATM LANs where there is usually plenty of available bandwidth (and thus congestion is usually not very severe).

3.4.1 The UBR service category in the standards

The first standard to support what became later the UBR service category was [For93]. In this document, the UBR traffic contract is similar to one of the CBR traffic contracts. When establishing a UBR VC, the only traffic related parameter that an endsystem has to specify is the Peak Cell Rate for the CLP=0+1 (PCR_{0+1}) cell flow for each direction of the VC. In addition to these traffic related parameters, the endsystem must also indicate that it requires a best-effort service for this VC.

While [For93] does not contain any particular requirement concerning the UPC for best-effort VCs, it is implicitly expected that no policing will be performed on UBR VCs. Furthermore, the PCR_{0+1} specified when establishing a best-effort VC is usually set to the physical line rate of the endsystem [PMH$^+$95] and in this particular case all the traffic sent by the endsystem on the best-effort VCs is by definition in conformance with the traffic contract.

In [For96c], the best-effort VCs were replaced by the UBR service category and two conformance definitions were specified. Both conformance definitions use the Peak Cell Rate for the CLP=0+1 cell flow (PCR_{0+1}) as their traffic contract, but they differ in the handling of the CLP bit by the network. The UBR.1 conformance definition corresponds to the best-effort VCs defined in UNI 3.0. It does not allow the network to modify the CLP bit of the cells on a UBR VC, while the UBR.2 conformance definition allows the network to tag any CLP=0 cell. The UBR.2 conformance definition was mainly introduced for some legacy switches which require the same explicit identification for the UBR cells as the low priority cells from VBR VCs. It should be noted that the tagging done in accordance with the UBR.2 conformance definition is not due to some non-conformance with some part of the traffic contract as in with VBR.3 conformance definition. The two UBR conformance definitions are summarized in table 3.4

Table 3.4: The UBR.1 and UBR.2 conformance definitions

Conformance definition	PCR_{0+1}	Network tagging
UBR.1	specified	not applicable
UBR.2	specified	applicable

3.4.2 The UBR service category and TCP/IP

When the first ATM Forum implementation agreement supporting the best-effort VCs appeared [For93], most ATM switches had a common characteristic. They used small buffers and buffers for a few tens or a few hundred cells per physical port were common. At that time, in most of the experimental ATM LANs, the main performance issue was how to utilize the large amount of bandwidth available in the ATM network since the available workstations and ATM adapters could only transmit and receive at a much lower rate than the line rate which connected them to the ATM network [MKK94]. In fact, when losses occurred, they mainly occurred at the receiving workstation and the ATM switches were rarely congested. However, some simulation studies revealed that if the workstations were able to saturate their ATM links, the overall performance of TCP in these ATM networks would be very low [RF94]. Two major causes for the low performance of TCP in such an environment were identified. The first problem was the small amount of buffers in the ATM switches. With a default packet size of 9188 bytes (192 ATM cells) for IP over ATM, an ATM switch with 256 cells of buffer per port had only enough memory to store one maximum-sized IP packet. This is not enough for the window-based congestion control mechanism used by TCP.

A second reason for the low performance of TCP in these ATM networks is the segmentation of a large TCP packet in a large number of ATM cells. When an ATM switch is congested and its buffers are full, it has to discard ATM cells. However, when the congestion is due to the fact that several large AAL5-PDUs are received at the same time in the switch, then the switch may have to discard several cells from each AAL5-PDU. This may create a waste of resources, since once a cell has been discarded from an AAL5-PDUs, all the cells that compose the entire AAL5-PDU are useless for the destination endsystem.

To minimize this problem, two simple modifications to the ATM switches have been proposed in [RF94]. The first modification, called Partial Packet Discard (PPD) requires the ATM switch to store some additional state information (one bit) for each VC. With PPD, when a switch has been forced to discard a cell from one AAL5-PDU due to an overflow of its buffers, it must continue to discard the remaining cells of this AAL5-PDU, except the last cell which is used for frame delineation. The main drawback of this solution is that cells from incomplete AAL5-PDUs are still transmitted uselessly in the network.

The second modification proposed in [RF94] is called Early Packet Discard (EPD). With EPD, an ATM switch tries to always discard entire AAL5-PDUs when it is congested. EPD works as follows. Each switch buffer uses a threshold whose value is smaller than the total buffer size. When the buffer occupancy is smaller than this threshold, all the arriving cells are accepted inside the buffer. When the buffer occupancy is above this threshold, the last cells of the "old" AAL5-PDUs are still accepted inside the buffer, but the cells from any new AAL5-PDU are discarded. Once a cell has been discarded from an AAL5-PDU, all the cells of this AAL5-PDU are discarded. A switch which implements EPD must maintain two bits of context memory for each VC.

The simulations reported in [RF94] have had a profound impact on the ATM switches used to carry TCP/IP traffic. Today, most ATM switches contain at least several thousands of cell buffers per port and some contain even more buffers. In addition, the EPD and PPD mechanisms are now very common and the latest ATM Forum signalling protocol [For96a] allows the endsystems to request the ATM switches to enable these mechanism on a per-VC basis.

Since then, a large number of researchers have studied the performance of TCP over ATM networks providing the UBR service category. Several other packet discard mechanisms [LST$^+$96a, WSR97, LNO96, HK98, RBL98] have been proposed to improve, notably, the fairness of EPD and PPD.

3.5 The ABR service category

The major drawback of the UBR service category is that it assumes that if congestion occurs, the upper layer protocols will be able to prevent a complete congestion collapse. While this assumption might be acceptable in lightly loaded and carefully managed networks where all the traffic is "controlled" by a protocol such as TCP which includes some congestion control mechanism, it is clear that this assumption is at least questionable when the upper layer protocols (e.g. UDP) do not include any congestion control mechanism or in public networks. To resolve this problem, the ATM Forum has worked, during several years, on the definition of a better best-effort service than the UBR service category. This new service class relies on a special congestion control mechanism which operates in the ATM layer.

Several congestion control mechanisms have been studied by the Traffic Management working group of the ATM Forum. Two main classes of mechanisms were proposed. The rate-based mechanisms [BF95] adapt the rate of transmission of the endsystems as function of the feedback received from the network. The credit-based mechanisms [KM95] rely on a window-based protocol which is used on each link to ensure that no cells are lost due to a buffer overflow in the switches or the endsystems. After many discussions, the Traffic Management working group of the ATM Forum finally opted for the rate-based mechanisms [BF95]. However, instead of a single congestion control mechanism, they chose to define a framework which supports different rate-based congestion control mechanisms. This congestion control framework is the main characteristic of the ABR service category.

3.5.1 The ABR rate-based congestion control framework

The ABR congestion control framework allows several congestion control mechanisms to be implemented. All these congestion control mechanisms have several common characteristics. The first one is that they operate in the ATM layer. The second one is that they modify the transmission rate of the endsystems in response to the feedback received from the network.

In the ABR congestion control framework, the feedback information is carried in special cells called the Resource Management (RM) cells. These cells are used to establish a feedback loop from the source endsystem to the destination endsystem and back via all the intermediate ATM switches. The flow of RM cells is shown graphically in figure 3.3.

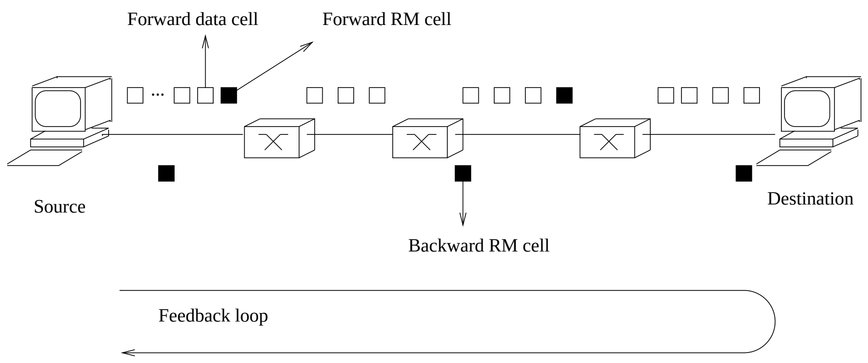

Figure 3.3: The flow of data and RM cells

The flow of RM cells is established as follows. The source endsystem sends one RM cell every NRM cells (the default value for NRM is fixed at 32 in [For96c]). These RM cells follow exactly the same path as the normal cells. When an ATM switch receives an RM cell, it may modify some fields of this cell in order to indicate its current level of congestion. When the RM cells reach the destination endsystem, they are sent back to the source endsystem, possibly after the modification of some fields of the RM cell if the destination endsystem itself is congested. These RM cells may again be modified by all the intermediate ATM switches on their way back to the source endsystem. The RM cells which are generated by the source endsystem are called forward-RM cells until they reach the destination endsystem and backward-RM cells when they are returning to the source endsystem.

The ABR congestion control framework supports two major modes of operation : the binary and explicit-rate modes. In binary mode, the network uses two bits to indicate whether it is congested or not and the endsystems adapt their transmission rates based on this feedback and according to predefined rules [For96c]. In explicit-rate mode, the network indicates in the RM cells the rate at which the endsystems should transmit. The ABR congestion control framework allows both modes to be used at the same time.

When operating in binary mode, a congested ATM switch may indicate its current congestion level in several ways. The first possibility is to set the Explicit Forward Congestion Indication (EFCI) bit in the header of the data cells on the VCs which are responsible for the current congestion. When the destination endsystem receives data cells it has to keep track of the value of the last received EFCI bit, and when it returns a backward RM cell to the source endsystem, it must set the Congestion Indication (CI) bit in this RM cell if the last received data cell had its EFCI bit set. A second possibility for an ATM switch[2] to indicate its congestion level is to modify the CI or No Increase (NI) bits on the forward or backward RM cells of the VCs that are responsible for the congestion.

The role of these two bits can be intuitively described as follows. The CI bit is set by

[2]This is also applicable for the destination endsystem when it is itself congested.

an ATM switch to indicate that it is congested and that the sources must reduce their transmission rate. The NI bit is set by an ATM switch to indicate that the sources should not increase anymore their rate, otherwise the switch will become congested. When a source endsystem receives a backward RM cell, it must examine the value of these two bits and possibly modify its transmission rate. If the CI bit is set, the endsystem must reduce its transmission rate. The rate decrease is performed according to equation 3.3, where the Rate Decrease Factor (RDF) is a parameter which is negotiated at connection establishment.

$$rate = rate * (1 - RDF) \tag{3.3}$$

If the CI bit is not set, then the rate modification depends on the value of the NI bit. If the NI bit is set, then the source should not modify its transmission rate. If the NI bit is not set, then the source is allowed to send at the rate given by equation 3.4 where the Rate Increase Factor is a parameter which is negotiated at connection establishment.

$$rate = rate + RIF * PCR \tag{3.4}$$

In addition to the CI and NI bits, the RM cells contain another important field called Explicit Rate (ER). When a source sends a forward RM cell, it must place in this parameter the rate at which it is willing to transmit[3]. All the switches (and the destination endsystem) must verify the ER field of all the in transit RM cells. If some intermediate switches or the destination endsystem cannot allow the VC to transmit at the rate indicated by the ER field, they should reduce it to a lower value. Upon reception of a backward RM cell, a source endsystem must first modify its transmission rate according to the value of the NI and CI bits (equation 3.3 or 3.4) and then it must take into account the value of the ER field (equation 3.5). This ensures that both binary and explicit-rate mode switches may coexist in the same network.

$$rate = min(min(rate, ER), PCR) \tag{3.5}$$

A complete description of the ABR congestion control framework, with a detailed explanation of the source, switch and destination behaviours would be outside the scope of this thesis, but the interested reader is referred to [BF95], [For96c] or [JKGF96].

3.5.2 The ABR traffic contract

The ABR traffic contract is the most complex traffic contract currently defined in [For96c]. It is composed of two types of parameters. The traffic related parameters are directly associated with the flow of ATM cells, but are independent of the ABR congestion. The other parameters are directly associated with the ABR congestion control mechanism.

The traffic-related parameters of the ABR traffic contract are :

- Peak Cell Rate (PCR) and associated Cell Delay Variation Tolerance (τ_{PCR})
- Minimum Cell Rate (MCR) and associated Cell Delay Variation Tolerance (τ_{MCR})

[3]It can be expected that this rate will often be equal to the PCR in practical implementations.

The PCR has the same meaning as with the other service categories. The MCR corresponds to the minimum amount of bandwidth which is reserved for the VC in the ATM network. When providing a classical best-effort service, the MCR shall be set to zero, but it may be set to a much higher value when some minimum bandwidth is required. When the MCR is larger than zero, the endsystem is assured of always being able to transmit at least at this rate, independently of the feedback received from the network[4].

The ABR congestion control mechanisms are characterized by several parameters whose value may be negotiated or signalled at connection setup. These parameters are :

- Initial Cell Rate (ICR)
- Rate Increase Factor (RIF)
- Rate Decrease Factor (RDF)
- Transient Buffer Exposure (TBE)
- Fixed Round-Trip Time (FRTT)
- NRM which fixes the frequency of the RM cells
- Time between RM cells (TRM)
- Allowed Cell Rate Decrease Time Factor (ADTF)
- Cutoff Decrease Factor (CDF)

The first four parameters are negotiated at connection establishment time between the endsystems and the network. The ICR is the rate at which the endsystem can transmit before receiving feedback from the network. The ICR value on each VC is a compromise between the endsystem's desire of a high ICR for bursty applications and the network desire of a low ICR to avoid uncontrolled transmission as much as possible. The RIF and RDF are used by the endsystems when they receive binary feedback from the network. The TBE corresponds to the maximum number of cells that the network is able to buffer during the startup phase of the VCs (i.e. before the RM loop has been established). The FRTT is a lower bound on the time required to receive an RM cell back at the source endsystem. It includes all the fixed delays (i.e. propagation and switching delays), but not the possible queueing delays. The FRTT is accumulated by all the switches on the path of a VC during connection establishment. It is used to compute notably the maximum ICR for the VC (after the establishment of the connection, the ICR is set to $ICR = max(ICR_{negotiated}, TBE/FRTT)$). The last four parameters have default values, but may optionally be negotiated. NRM is the number of cells that an endsystem may send for each forward RM cell. Its default value is 32. TRM provides an upper bound on the time between two forward RM cells. It implies that, when it is active, a source must send at least one forward RM cell every TRM seconds. The ADTF and CDF are used to define how the endsystem shall decrease its transmission rate down to ICR if the flow of backward RM cells is interrupted at some time.

[4]With a non-zero MCR, equation 3.5 becomes $rate = min(max(MCR, min(rate, ER)), PCR)$, provided that $MCR \leq PCR$.

3.5.3 The ABR conformance definition

As with the other service categories, a conformance definition is necessary for the ABR traffic contract. For the ABR traffic contract, the conformance definition is composed of two parts. The first part is the conformance to the $PCR, CDVT_{PCR}$ part of the traffic contract. This part is defined, as usual, by GCRA$(1/PCR, CDVT_{PCR})$. The second, and most important, part of the conformance definition defines the conformance of the cell flow sent by an endsystem with the feedback received from the network. This conformance definition relies on the Dynamic-GCRA (D-GCRA). In the standard GCRA, the conformance of the cell flow is verified by comparing the arrival time of each cell with a theoretical arrival time which is incremented by the peak emission interval for each conforming cell. The D-GCRA is derived from the GCRA. The main difference between the GCRA and the D-GCRA is that in the D-GCRA the "theoretical arrival time" is incremented by a "current emission" interval which reflects the feedback received from the network. This "current emission interval" may potentially change for every ATM cell. We will use I(k) in the remainder of this section for the "current emission interval" for the k^{th} ATM cell. When I(k) is constant, the D-GCRA is equivalent to the GCRA.

Several algorithms have been proposed to compute I(k). These algorithms update I(k) to reflect the backward RM cells received from the network. The update of I(k) must take into account the delay between the location of the conformance definition and the endsystem. For example, let us consider the situation depicted in figure 3.4. In this case, there is a non-zero propagation delay d between the location of the conformance definition and the ABR source. This delay implies that the D-GCRA must wait some time before applying a rate increase or rate decrease received in a backward RM cell.

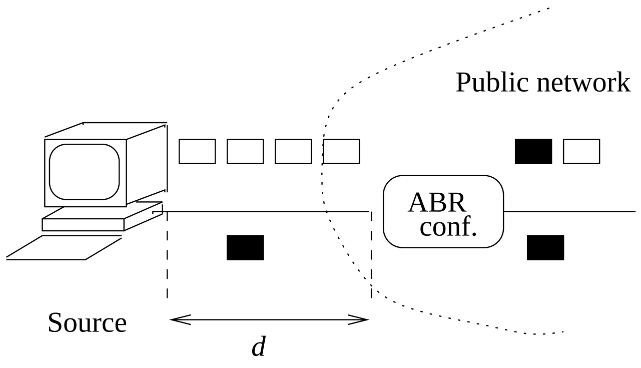

Figure 3.4: Example configuration with ABR conformance

To understand the impact of this non-zero delay, let us consider that a source is transmitting at rate R and that a backward RM cell instructs the source to transmit at a rate $R/2$ (figure 3.5). This backward RM cell is first received by the conformance definition. If the conformance definition used immediately the lower rate in the D-GCRA, then some cells would be declared non-conforming. This is not desirable since the source is behaving correctly. To cope with the propagation delay between the conformance definition and the source and the time to process the RM cell by the source, the conformance definition will wait some time before using the lower rate in its D-GCRA. The standardization bodies have chosen to use two different delays in this case : τ_2 in case of a rate decrease and τ_3

in case of a rate increase. The reader interested in the detailed algorithms to compute I(k) is referred to [For96c] and [ITU96c].

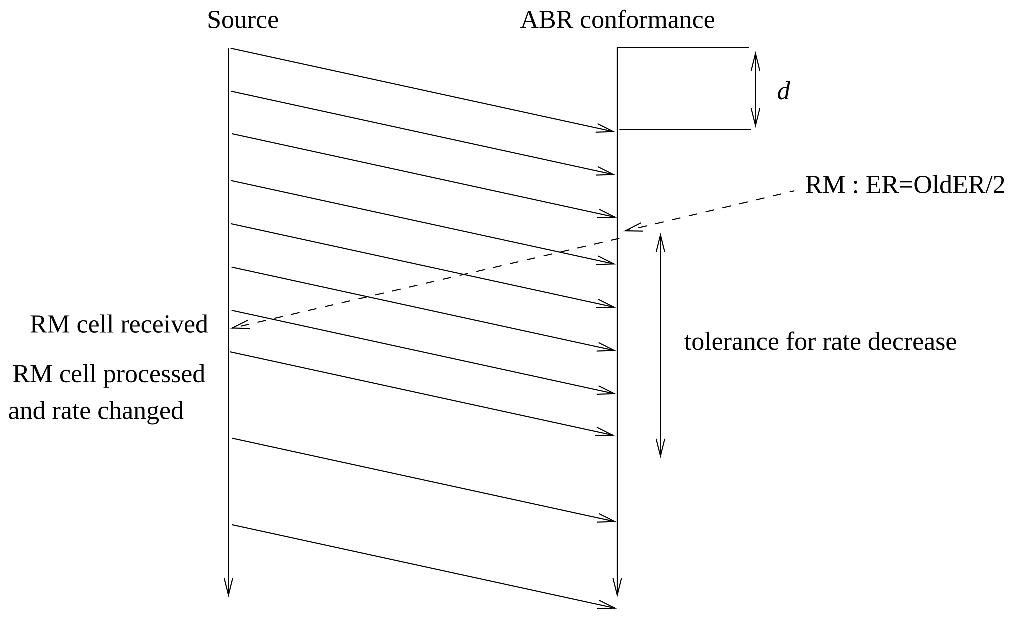
Source
ABR conformance
d
RM : ER=OldER/2
RM cell received
RM cell processed and rate changed
tolerance for rate decrease

Figure 3.5: Impact of a non-zero delay between the conformance definition and the ABR source

An important point to note concerning the ABR conformance definition and the computation of $I(k)$ is that the algorithms specified in [For96c] and [ITU96c] only utilize the ER field of the received backward RM cells to compute $I(k)$.

3.5.4 Virtual Source/Virtual Destination

In addition to the mode where there is a single ABR control loop between the source and the destination endsystem, the ABR specification also supports Virtual Sources (VS) and Virtual Destinations (VD). These VS and VD implement the source and destination endsystem specification defined in [For96c], but inside ATM switches. By using these VS and VD, a network operator can break long ABR control loops to improve their efficiency or to isolate different parts of his network (figure 3.6). A VD receives the forward RM cells and returns them as backward RM cells and transmit the data cells to its corresponding VS in the same switch. A VS transmits the data cells received from the corresponding VD, generates forward RM cells and shapes its cell flow according to the feedback received in the backward RM cells. In addition, there are some interactions between the corresponding VS and VD in the same switch so that if a VS becomes congested the corresponding VD may reduce the ER in the backward RM cells returned to the upstream nodes.

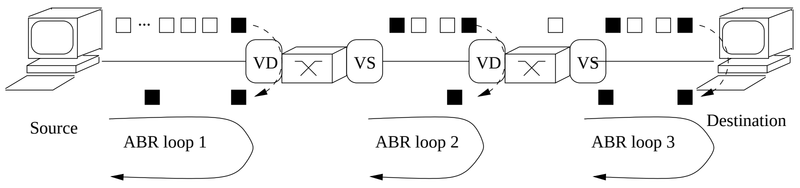

Figure 3.6: ABR Virtual Source/Virtual Destination

3.6 The GFR service category

The Guaranteed Frame Rate (GFR) service category is one of the most recent ATM service categories. It was first proposed in December 1996 [GH96] and became quickly very popular. GFR is now being actively studied by both the ATM Forum and the ITU-T. It can be expected that the GFR service category will be part of the next releases of the ATM Forum Traffic Management specification and of the ITU-T I.371 recommendation.

The main motivation for the introduction of this new service category was to provide a service which is as easy to use as the UBR service category for the endsystems while still providing bandwidth guarantees. In [GH96], the promoters of the GFR service category have argued that for most deployed endsystems, the only really usable service category defined in [For96c] was the UBR service category because these endsystems either are directly connected to the ATM network, but with an ATM adapter which does not provide the shaping mechanisms required by the CBR, VBR and especially ABR service categories or are not attached directly to the ATM network. In the latter case, which corresponds to most of today's corporate networks, the endsystems are fitted with Ethernet or Token-Ring adapters and are connected through intermediate systems (i.e. bridges or routers) to the ATM network. For these endsystems, the guarantees offered by the ATM network are completely hidden by the intermediate system.

The GFR service category keeps the simplicity of the UBR service category (from the endsystem's point of view) by allowing the endsystem to always transmit cells at the line rate of their ATM adapter. Another important feature of the GFR service category is that since almost all the data traffic is AAL5-based, GFR takes the specific requirements of AAL5 into account. The GFR service category requires the network elements to be aware of the AAL5 PDU boundaries and to discard entire AAL5-PDUs when congestion occurs. Such a strong requirement was not included in the UBR service category, although it is also mainly used for AAL5-based traffic. Another important difference between the GFR service category and the UBR service category is that the GFR service category allows the user to reserve some bandwidth, for each GFR VC, inside the network. This means that the user is assured that he will always be able to transmit at a minimum rate without losses. On the other hand if the network is not congested, the user will be able to transmit at a higher rate. As discussed in chapter 2, the provision of such a minimum bandwidth is very useful in several situations.

More precisely, the GFR traffic contract [Ken98] is composed of four main parameters :

- Peak Cell Rate (PCR) and associated Cell Delay Variation Tolerance (τ_{PCR})

- Minimum Cell Rate (MCR) and associated Cell Delay Variation Tolerance (τ_{MCR})
- Maximum Burst Size (MBS)
- Maximum Frame Size (MFS)

The PCR has the same meaning as with the UBR service category : it is the maximum rate at which the endsystem is allowed to transmit. It can be expected that the PCR will often be set at the line rate of the ATM adapter of the endsystems. The MFS is the largest size of the AAL5-PDUs that the endsystems can send. For GFR SVCs, this parameter will be equal to the AAL5-CPCS SDU size parameter which is negotiated between the source and destination endsystems during connection setup [For96a].

The endsystems request a minimum guaranteed bandwidth by specifying a non-zero MCR and an associated MBS. The MCR, expressed in cells per second, corresponds to the long term average bandwidth which is reserved for the VC inside the network. It is similar to the Sustainable Cell Rate (SCR) used with the VBR service category [For96c], although the MCR provides a minimum guaranteed bandwidth to entire AAL5-PDUs while the SCR provides a minimum guaranteed bandwidth to individual cells. The MBS places an upper bound on the burstiness of the traffic to which the minimum guaranteed bandwidth applies. The value of the MBS is negotiated between the endsystems and the network, but this parameter must always be at least equal to the MFS.

The GFR service category defines a particular utilization of the CLP bit in the ATM cells. With the GFR service category, the endsystem is allowed to transmit either CLP=0 AAL5-PDUs[5] or CLP=1 AAL5-PDUs. The CLP=1 AAL5-PDUs are considered as low priority AAL5-PDUs which should be transmitted by the network on a best-effort basis. The minimum guaranteed bandwidth is not applicable for CLP=1 AAL5-PDUs and these AAL5-PDUs should be discarded earlier than the CLP=0 AAL5-PDUs when congestion occurs.

With this utilization of the CLP bit, the meaning of the MCR is that if the endsystem transmits CLP=0 AAL5-PDUs at a rate smaller or equal to the MCR, then all these AAL5-PDUs should be correctly received by the destination. However, the GFR service category does not require the endsystems to shape their traffic and it can be expected that most users of this service category will always transmit at the negotiated PCR. In this case, each AAL5-PDU will appear as a burst of cells transmitted at the PCR. The MBS parameter of the GFR traffic contract is used to support this bursty behaviour.

Formally, the minimum guaranteed bandwidth is specified by F-GCRA(T,f) [Ken98] with parameters $T = 1/MCR$ and $f \geq \tau_{MCR} + (MBS - 1) \times (1/MCR - 1/PCR)$. The Frame-Generic Cell Rate Algorithm (F-GCRA) (figure 3.7) is an adaptation of the GCRA used with the VBR service category. The main difference between the GCRA and the F-GCRA is that the F-GCRA declares entire AAL5-PDUs to be eligible or non-eligible for the minimum guaranteed bandwidth. The eligible AAL5-PDUs are those which should be delivered to the destination to fulfill the minimum guaranteed bandwidth. Only the CLP=0 AAL5-PDUs can be declared eligible for the minimum guaranteed bandwidth by the F-GCRA. While the F-GCRA is used to specify which AAL5-PDUs are eligible for

[5]A CLP=0 AAL5-PDU is an AAL5-PDU composed of CLP=0 cells. The GFR service category does not allow the endsystems to transmit AAL5-PDUs containing both CLP=0 and CLP=1 cells.

the minimum guaranteed bandwidth, it should be noted that the GFR service category explicitly allows the endsystems to transmit AAL5-PDUs in excess of this minimum guaranteed bandwidth. The GFR service category also expects the network to deliver this excess traffic on a best-effort basis to the destination endsystems and to "fairly" distribute the available bandwidth to the active VCs.

```
Cell Arrival at time $t_a$ :
 First cell of an AAL5-PDU:
 if( ( $t_a < TAT - f$ ) OR (IsCLP1(cell))
 {
 /* non-eligible cell */
 eligible=FALSE;
 }
 else
 {
 /* eligible cell */
 eligible = TRUE;
 $TAT = max(t_a, TAT) + T$;
 }

 Middle or last cell of an AAL5-PDU :
 if(eligible)
 {
 /* eligible cell */
 $TAT = max(t_a, TAT) + T$;
 }
 else
 {
 /* non-eligible cell*/
 }
```

Figure 3.7: F-GCRA(T,f)

As with other service categories (e.g. VBR), two conformance definitions have been defined for the GFR service category : GFR.1 and GFR.2. The only difference between the two conformance definitions is whether a F-GCRA is used to tag the non-eligible AAL5-PDUs at the ingress of the network or not[6].

With the GFR.2 conformance definition, the UPC function at ingress of the network uses a F-GCRA to tag the non-eligible AAL5-PDUs. When this conformance definition is used, only the eligible AAL5-PDUs are accepted as CLP=0 AAL5-PDUs inside the network. Thus, there is a clear distinction between the eligible (CLP=0) and the non-eligible (CLP=1) AAL5-PDUs and the ATM switches may rely on this to decide whether an AAL5-PDU must be delivered to fulfill the minimum guaranteed bandwidth or not. As we will see in chapter 7, a simple switch implementation can be used to support the GFR.2 conformance definition.

With the GFR.1 conformance definition, the network is not allowed to modify the CLP bit of the AAL5-PDUs sent by the endsystems[7], but the endsystems are still allowed to send CLP=0 AAL5-PDUs in excess of the minimum guaranteed bandwidth (even if only a fraction of these AAL5-PDUs are actually eligible for the guaranteed minimum bandwidth). With the GFR.1 conformance definition, there is thus no "visible" distinction between an eligible and a non-eligible AAL5-PDU inside the network. Thus, to support the GFR.1 conformance definition, each ATM switch in the network must be able to determine, by itself, which CLP=0 AAL5-PDUs must be transmitted to fulfill the minimum guaranteed bandwidth and which AAL5-PDUs are part of the excess traffic and thus could be discarded if congestion occurs.

[6]The details of some parts of these conformance definitions are still being discussed within the ITU-T and the ATM Forum [Ken98, BD98, HV98b], but the F-GCRA and difference between GFR.1 and GFR.2 are now stable. We will only consider this part of the GFR conformance definition in this thesis.

[7]This means that the UPC does not use a F-GCRA with the GFR.1 conformance definition.

The eligible AAL5-PDUs are those which must be delivered to the destination to fulfill the minimum guaranteed bandwidth. However, it should be noted that the GFR service category does not strictly require that the eligible AAL5-PDUs are exactly those which must be delivered to the destination to provide the minimum guaranteed bandwidth. The requirement is weaker. The GFR service category only requires the network to deliver enough entire CLP=0 AAL5-PDUs at the destination to provide the minimum guaranteed bandwidth, but it does not specify precisely which CLP=0 AAL5-PDUs must be delivered to the destination.

3.7 Other service categories

In addition to the five service categories discussed in the previous sections, the ITU-T has standardized another service category called ABT, and is currently defining another service category called CT. We will briefly describe these service categories in sections 3.7.1 and 3.7.2. However we will not consider these service categories in this thesis since, up to now, ABT has not yet received a lot of attention from the industry and CT only provides a best-effort service.

3.7.1 ATM Block Transfer

ATM Block Transfer (ABT) has its roots in the fast reservation protocols developed at CNET in the early 1990s [BT92]. With the CBR and VBR service categories, the signalling protocols have two roles. When a connection is established, they create a path between the source and destination endsystem, but they also reserve the bandwidth associated with the VC for its whole lifetime. ABT decouples these two roles. With ABT, the signalling protocol is still used to create the path between the source and destination endsystems but does not explicitly reserve bandwidth. With ABT, the actual bandwidth reservation is performed with a special protocol implemented with RM cells [ITU96c].

ABT assumes that the endsystems transmit blocks of data, where a block contains data cells and is delimited by RM cells [Gui97]. Two variants of ABT have been defined by ITU-T : ATM Block Transfer with Delayed Transmission (ABT/DT) and ATM Block Transfer with Immediate Transmission (ABT/IT). With ABT/DT, when an endsystem wishes to transmit a new block of data, it first sends a request to the network with a special RM cell. This RM cells indicates the requested transmission rate for this block. The network will examine this request and will forward it to the destination endsystem, possibly indicating a lower rate. It should be noted that the destination endsystem is not involved in the negotiation of the new transmission rate. The network will then acknowledge the request and authorize the source endsystem to transmit at the rate negotiated between the source endsystem and the network by sending a special RM cell to the source endsystem. Upon reception of this cell, the source endsystem will first send a RM cell to acknowledge this rate modification and then will transmit its block at the negotiated rate. ABT/DT also allows the network or the destination endsystem to reduce the transmission rate inside a block by sending a special RM cell to the source endsystem. At the end of a block, the source endsystem must send a RM cell indicating a rate of zero to inform the network and the destination endsystem of the end of the block.

ABT/IT uses a more optimistic approach. With ABT/IT, the source endsystem still transmits blocks of data, but the transmission rate of each block is not negotiated between the source endsystem and the network. When a new block starts, the source endsystem sends a special RM cell to indicate the transmission rate for this block and the transmission of the block immediately follows this RM cell. If there are enough resources available in the network to transmit the block, the entire block will be delivered at the destination. Otherwise, the entire block will be discarded.

The ABT/DT and ABT/IT traffic contracts are composed of four parameters :

- Peak Cell Rate (PCR_{0+1}) and associated Cell Delay Variation Tolerance for the CLP=0+1 cell flow ($\tau_{PCR_{0+1}}$)
- Sustainable Cell Rate (SCR_{0+1}) and associated Cell Delay Variation Tolerance for the CLP=0+1 cell flow ($\tau_{SCR_{0+1}}$)
- Burst tolerance for the CLP=0+1 cell flow ($\tau_{IBT_{0+1}}$)
- Peak Cell Rate (PCR_{RM}) and associated Cell Delay Variation Tolerance for the RM cell flow ($\tau_{PCR_{RM}}$)

The Peak Cell Rate is the highest rate at which a block can be transmitted. The Sustainable Cell Rate can be used to reserve some bandwidth for an ABT VC. A non-zero SCR_{0+1} allows the endsystems to reserve a minimum amount of bandwidth for each ABT VC. The meaning of this minimum bandwidth is that on average the endsystems will be able to utilize this bandwidth. The Burst tolerance is used to indicate the maximum burstiness of the traffic. The PCR for the RM cell flows corresponds to the maximum rate at which the endsystem will be allowed to perform rate negotiation through the use of RM cells.

Up to now, the ABT service category has not been very popular within the ATM community. While ABT is specified in the ITU-T recommendations, it has never been accepted by the ATM Forum and we do not expect ABT to become widely used or even implemented in the near future.

3.7.2 Controlled Transfer

A lot of work on credit-based flow control mechanisms has been performed while the ATM Forum was working on the ABR service category. After many discussions, the ATM Forum has opted for a rate-based flow control mechanism and the work on the credit-based mechanisms has stopped within the ATM Forum. However, several companies have continued to work on this subject and two different credit-based mechanisms [All97] [LBC97] were recently proposed within ITU-T. After some discussions, the proponents of these two credit-based mechanisms have agreed to submit a common proposal called Controlled Transfer (CT).

The main application for CT will be to provide a loss-free best effort service. CT provides this service by using a credit-based mechanism on all the links to prevent losses inside the switch buffers. Since CT provides a best effort service, it is not suitable to provide services with a minimum guaranteed bandwidth. Furthermore since it is only accepted by the ITU-T and not the ATM Forum, we do not expect this new service category to be widely supported by equipment vendors.

3.8 Conclusion

In this chapter, we have described in details the service categories which have been defined by the standardization bodies for ATM network. We have explained the characteristics of each service category and described its evolution within the standardization bodies.

We have shown that two types of service categories have been defined by the standardization bodies. With the open-loop service categories (CBR, VBR, UBR and GFR), the network does not provide any explicit feedback to the endsystems. These service categories either avoid congestion by limiting the amount of traffic which is accepted inside the network (e.g. CBR and VBR.1) or discard cells (VBR.2/3 and UBR) or entire AAL5-PDUs (UBR and GFR) if congestion occurs. Since the first ITU-T recommendation on traffic control [ITU93c], the open-loop service categories have clearly evolved to better support AAL5 based traffic. This evolution is summarized in table 3.5.

Table 3.5: The evolution of the open-loop service categories

	CBR	VBR	UBR	GFR
ITU-T I.371v1	yes	no	no	no
ATM Forum UNI 3.0	yes	yes	yes	no
ATM Forum TM 4.0	yes	yes	yes	no
ITU-T I.371v2	yes	yes	no	no
ATM Forum TM 5.0	yes	yes	yes	*expected*
ITU-T I.371v3	yes	yes	no	*expected*

Table 3.6: The evolution of the closed-loop service categories

	ABR	ABT	CT
ITU-T I.371v1	no	no	no
ATM Forum UNI 3.0	no	no	no
ATM Forum TM 4.0	yes	no	no
ITU-T I.371v2	yes	yes	no
ATM Forum TM 5.0	yes	no	no
ITU-T I.371v3	yes	yes	*expected*

The closed-loop service categories (ABR, ABT and CT) are more recent. With these service categories, the network provides an explicit feedback to the endsystems by using special RM cells. For the three closed-loop service categories, this feedback aims at avoiding cell losses inside the network. However, as explained in this chapter, the three closed-loop service categories use completely different feedback mechanisms. Table 3.6 summarizes the current support for the closed-loop service categories in the standards.

Shortly before we finished this thesis, a new service category, called Weighted Unspecified Bit Rate (UBRw), was proposed within the ATM Forum [KDW$^+$98]. This new service category was proposed to support different relative drop priorities for best-effort VCs. The UBRw proposal was not accepted as a new work item by the ATM Forum members who consider that GFR is sufficient to efficiently support TCP/IP traffic in ATM networks.

Chapter 4

TCP and the CBR service category

4.1 Introduction

Since the CBR service category was the first to be defined by the ITU-T, it is not suprising that it was also the first service category to be supported in public networks. In this chapter, we present and analyze detailed measurements with TCP/IP in a wide area network which offered a CBR VPs.

This chapter is structured as follows. We first present the characteristics of our measurement environment in section 4.2. We then analyze the results of a first campaign of measurements in sections 4.3 and 4.4. Based on the results of these measurements, we performed a second campaign of measurements to explore the influence of the IP MTU size on the performance of TCP/IP. This second campaign of measurements is discussed in sections 4.5 and 4.6. We then study the impact of possible improvements to TCP and the CBR service category in sections 4.7 and 4.8. Finally, we look at the impact of the lessons learned from our measurements on the signaling and management protocols in section 4.9 and discuss related work in section 4.10.

4.2 Measurement environment

In January and August 1995, we had the opportunity to evaluate the performance of TCP in an experimental wide area network supporting the CBR service category. This performance evaluation was performed in two campaigns of measurements that allowed us to gain a better understanding of the interactions between a transport protocol such as TCP and the CBR service category. Those measurements were performed with equipments that were commercially available at that time. Since then, the ATM equipments have continued to evolve, but the lessons learned from these measurements still hold. Before discussing these measurements in details, we will briefly present the characteristics of our measurement environment.

Our measurement environment consisted of ATM LANs interconnected via a wide area ATM network (the European ATM Pilot).

4.2.1 The European ATM Pilot

The European ATM Pilot [PRW+95] is one of the first public wide area ATM networks deployed in the world. This network has been built as a collaboration among 16 Public Network Operators (PNOs) from 15 countries, and covers most of western Europe. Each PNO participating in the Pilot network has installed at least one ATM switch and established links with its adjacent countries. Wide area ATM switches from different vendors are used inside the network. These ATM switches are entirely managed by the PNOs, and as simple users of the network, we had no control on this equipment. Most of the links on the European ATM Pilot are 34 Mbps E-3 ATM links, but some links have already been upgraded to 155 Mbps SDH links.

The measurements presented in this chapter were done between the ATM LANs of the University of Liège (Belgium), Electricité de France (Paris, France) and Telenor Research and Development (Kjeller, Norway). These ATM LANs were connected to their national ATM network with links provided, respectively by Belgacom, France Telecom and Telenor. We also used transit links provided by Deutsche Telekom (Germany) and Telia (Sweden).

The European ATM Pilot network provided a VP bearer service. As the network did not support signaling or virtual circuits when we performed our measurements, we used semi-permanent VPs. Those semi-permanent VPs were activated during short periods of time (usually half a day). The European ATM Pilot only supported the CBR service category, and when we requested the activation of a VP, we had to specify the PCR (PCR_{0+1}) of our traffic. While the three testbeds used for the measurements had at least a 34 Mbps (80000 cells per second) link to the European ATM Pilot, the PNOs did not allow us to use VPs with a bandwidth equal to our access link. That is the reason why we used VPs with a bandwidth of approximately 27 Mbps (61000 cells per second).

In order to protect the network from misbehaving users, and in accordance with the ATM Forum specifications [For93] and ITU-T recommendations [ITU93c], a UPC mechanism was used in the European ATM Pilot. As our traffic contract specified PCR_{0+1}, the tagging option was not applicable for this UPC. The provision of a CBR service, combined with the usage of a UPC was probably one of the main characteristics of the European ATM Pilot compared with other experimental ATM networks.

4.2.2 The ATM LANs

The three ATM LANs we used were very similar. They were all built around an ASX- 200 ATM switch from Fore Systems Inc. The ASX-200 is a 2.5 Gbps bus-based, non- blocking, output buffered switch equipped with an internal Sparc 2 based controller. Each output port contains a buffer which can hold 256 cells. The UPC mechanism available in the ASX-200 was disabled during the measurements. The testbeds located in Liège and Paris were connected with an E-3 link to their national ATM Pilot, while the testbed located in Oslo used an OC-3 link.

The workstations used in the three testbeds were Sparc 10 clones from Axil (model 311/5.x) running SunOS 4.1.3_U1, and equipped with an SBA-200 ATM adapter from Fore. The testbed located in Liège used SDH 155 Mbps SBA-200 ATM adapters, while the other testbeds used Taxi 100 Mbps and Taxi 140 Mbps ATM adapters. These variants of the SBA-200 offer similar performance. Measurements done in the local area have shown that the bottleneck on the SBA-200 is not the physical rate of the ATM link, but mainly the transfers between the workstation memory and the ATM adapter [MKK94]. The maximum TCP throughput measured by ttcp [SM84] with these SBA-200 ATM adapters on our workstations are similar. For the two campaigns of measurements presented in this chapter, the sender was always located in Liège.

4.2.3 Characteristics of a conforming traffic

In a wide area network such as the European ATM Pilot, where the traffic contract is enforced by a UPC mechanism, it is essential for the user to generate a conforming traffic. If the traffic is not completely conforming, the UPC will discard the non-conforming cells, and this may lead to a high cell loss rate.

In our environment (figure 4.1), cells can be discarded in two places. The local ATM switch will loose cells if there is an overflow in its (256 cells) output buffer, and the UPC in the ATM Pilot will discard cells that are not in conformance with the traffic contract.

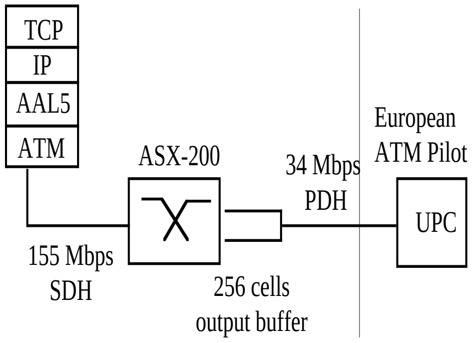

Figure 4.1: Sending side of our environment for the first campaign of measurements

The maximum burst size sent at a 155 Mbps line rate which does not cause an overflow of the output buffer on the 34 Mbps port of the ASX-200 is given by equation (4.1), assuming that the output buffer is idle before the transmission of the burst. That is, if T_i is the cell time on the input link of the switch, T_o the cell time on the output link of the switch, and L the size of the output buffer, measured in cells, the maximum burst size (in cells), neglecting the physical layer framing, is :

$$maximum\_burst = \frac{L \times T_o}{T_o - T_i} \tag{4.1}$$

In our sending environment (T_i=2.726 μsec, T_o=12.50 μsec, L=256 cells), the maximum burst size is thus 327 cells. Such a burst has to be followed by an idle time of $256 * 12.50 = 3200 \mu sec$ (i.e. 1174 cell times at the 155 Mbps line rate) to empty the output buffer of the switch.

However, a burst absorbed by the ASX-200 must still be conforming with the traffic contract enforced by the UPC mechanism. If T_j is the cell time at the input line of the UPC, T_u and τ_u respectively the peak emission interval and the cell delay variation enforced by the UPC, from equation 3.1, the maximum burst size declared as conforming by the UPC, assuming that the UPC is in the idle state upon the arrival of the first cell, is :

$$maximum\_burst = 1 + \frac{\tau_u}{T_u - T_j} \tag{4.2}$$

For example, let us consider that the traffic contract enforced by the UPC mechanism is 61000 cells/sec (one cell every 16.39 μsec), with an allowed Cell Delay Variation (CDV) of 101 μsec ([T=16.39 μsec, τ=101 μsec]). In our environment (T_j=12.50 μsec, T_u=16.39 μsec, τ_u=101 μsec), the maximum burst size declared as conforming by the UPC is only 26 cells. This burst has to be followed by an idle time of at least 104 μsec to let the UPC return to the idle state. Thus, the most bursty traffic allowed by the [T=16.39 μsec, τ=101 μsec] UPC is composed of bursts of 26 cells separated by an idle time of 104 μsec (i.e. 38 cell times at the 155 Mbps line rate).

As shown by equation 4.2, the maximum burst size declared as conforming by the UPC depends on both the cell rate of the traffic contract and the cell rate at the input of the UPC.

4.2.4 The ATM LANs and the CBR traffic contract

To fulfill the CBR traffic contract, we had to introduce some traffic shaping capabilities in our ATM LANs. As the VP established through the European ATM Pilot was terminated on the ASX-200, the best solution would have been to use a traffic shaper on the output port of the ASX-200 or an external traffic shaper on the E-3 link. Unfortunately, when we performed our measurements, the ASX-200 did not offer any traffic shaping capabilities[1], and we did not have an external traffic shaper for the first campaign of measurements. As we only used one VC inside this VP and there was no other traffic on the VP, another possibility was to perform the traffic shaping at the endpoints of the VC, i.e. on the ATM adapters.

Like most first generation ATM adapters [Dav93] [TS93], our Fore SBA-200 ATM adapters did not contain any custom chip to perform the traffic shaping on a per-VC basis. However, the SBA-200 [CMSB91] includes an i960 microprocessor that is mainly used for the data movements between the adapter and the host memory and the AAL segmentation and reassembly (with a hardware assisted CRC computation). Thanks to this i960, the SBA-200 is flexible, and by modifying the firmware of the i960, Fore Systems was able to integrate some traffic shaping capabilities on their adapter without a hardware upgrade. Fore Systems provided us with two firmware releases supporting traffic shaping. The first one was an unsupported modified version of the 2.2.9 firmware, and the second one was a supported 2.3.0 release. The modified 2.2.9 firmware shaped the traffic on a per-port basis by inserting idle cells between assigned cells, and thus was unable to efficiently shape several VCs simultaneously [KK95]. To have a better understanding of these two firmwares, the following experiment was performed at Telenor. One workstation, equipped with a Taxi 140 Mbps SBA-200 sent a train of 8 KBytes (i.e. 172 cells) long UDP packets through an ASX-200 ATM switch, to an Alcatel ATGA ATM tester equipped with a 155

[1]A very limited traffic shaping capability (one VP per port) is supported by the type "C" network modules on the ASX-200 [Sys95], but we received these network modules when we were not anymore connected to the European ATM Pilot

Mbps SDH interface. The ATGA measured the cell inter-arrival time (IAT), i.e. the difference measured in cell times between two successive assigned cells, of the ATM cells in the UDP packets. The requested throughput at the ATM level was one cell every 17.35 μsec (i.e. one cell every 6.4 cell slots on the 155 Mbps SDH interface of the ATGA).

Figure 4.2 shows that release 2.2.9 produces an almost regular traffic. The only significant variation in the IAT is an increase of 7-8 cell times[2] every 21 or 22 cells. This figure also shows that the IAT between two cells of consecutive packets jumps to 25 cell times, and thus the inter-packet time can be estimated as approximately 20 cell slots or 52 μsec. According to the documentation supplied by Fore Systems, this modified firmware is able to insert an idle time of n *0.492 μsec between two consecutive assigned cells transmitted on the output link. However, detailed measurements of the spacing [KK95] have shown that, on average, this idle time was much closer to n*0.473 μsec. We will use this measured value throughout this chapter. The value n, and thus the average rate of the output link, can be changed during run-time.

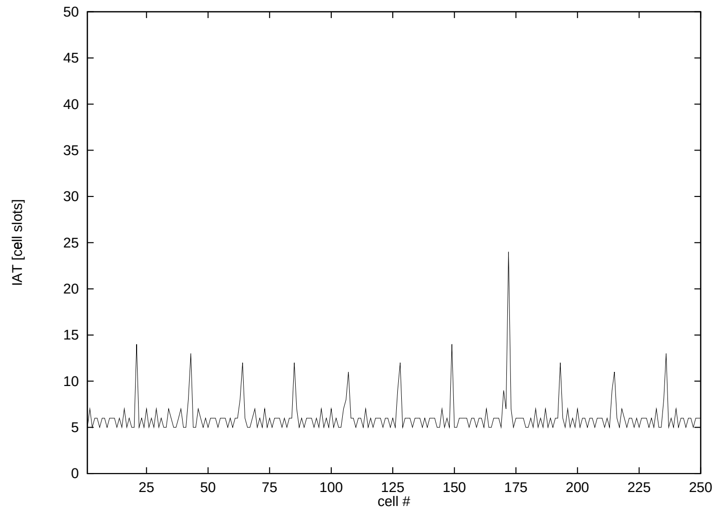

Figure 4.2: Spacing with release 2.2.9 of the Fore driver

The measurements with firmware 2.3.0 revealed that it transmitted bursts[3]of 6 cells at the line rate with an inter-burst time of almost 40 cell times. The average inter-arrival time was 6-7 cell times. As we had no a priori information on the amount of CDV tolerated by the UPC of the European ATM Pilot, we choose to use firmware 2.2.9 for our measurements as it produced the most regular traffic.

[2]As the measurements were taken on the 155 Mbps port of the ATGA, one cell time corresponds to 2.726 μsec.

[3]We learned later from Fore Systems Technical Support that the duration of these bursts was "configurable" by modifying some kernel variables with a debugger, but we were not aware of this feature when we performed our measurements.

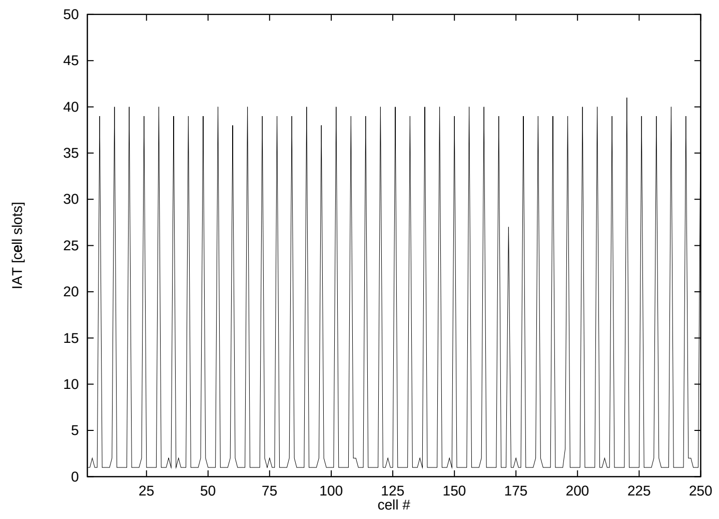

Figure 4.3: Spacing with release 2.3.0 of the Fore driver

4.3 First campaign of measurements

For this first campaign of measurements, we used the testbeds located in Liège and Paris. The distance between these two testbeds is approximately 400 km, and the round- trip-time measured by "ping" is 8 msec.

For these measurements, we requested a 58762 cells per second (i.e. one cell every 17.02 μsec or 24.9 Mbps) VP from the European ATM Pilot. This VP was monitored by a UPC inside the network, and we were told later that this UPC was set to enforce a traffic contract of 61000 cells per second (i.e. one cell every 16.39 μsec) with an allowed CDV of 101 μsec. Each TCP measurement consisted of the memory-to-memory transfer of 8 megabytes between the sender and the receiver with ttcp [SM84]. The workstations, the local ATM switches, and our VP in the ATM Pilot were exclusively used for the measurements. Thus, there were no contentions due to the sharing of the VP by several TCP connections.

The results showed that the TCP throughput is very sensitive to segment losses. In the following sections, we will present results from individual measurements which are representative of the problems we encountered during these measurements.

4.3.1 Measurements without traffic shaping

A common belief within the data communication community, based on TCP's excellent track record, is that TCP can adapt to any kind of bandwidth. However, during congestion and in case of traffic contract violations in ATM networks, the network will discard ATM cells, but not complete TCP segments. This may result in a flow of ATM cells but no flow of correct TCP segments, and thus no real data-transfer. We tried to perform some measurements without enabling the traffic shaping on our ATM adapters, and it was impossible to transfer any data with TCP. The TCP connection was easily established (the SYN segments and the corresponding ACKs are contained in a single ATM cell). However, it was impossible to perform any data transfer. This problem was caused by the default Maximum Transmission Unit (MTU) size of 9188 bytes selected for IP over ATM[Atk94]. When spacing is not enabled, the ATM adapters send the AAL-PDUs containing TCP segments as bursts of 192 cells, at the line rate. While a single burst of 192 cells can be absorbed by the output buffer of the local ATM switch (equation 4.1), the resulting burst on the 34 Mbps link is not conforming with the traffic contract, and at least one cell from each TCP segment containing data is discarded by the UPC of the ATM Pilot. Thus, with the default MTU size for IP over ATM, ATM level traffic shaping was really necessary, even with TCP, in our environment.

4.3.2 Measurements with a conforming traffic

With our SBA-200 ATM adapters, the closest possible value for the cell spacing, while still remaining larger than the peak emission interval of the traffic contract was one cell every 16.44 μsec. We performed measurements with TCP windows of 16 KBytes, 32 KBytes and 48 KBytes and with Transport Service Data Unit (TSDU) sizes[4] ranging from 1 KBytes to 16 KBytes. As shown in figure 4.4, the highest throughput is obtained with the largest window size. Measurements with the spacing set to a value higher than one cell every 16.44 μsec gave similar results. The throughput variations shown in figure 4.4 are caused by limited segment loss (the segment loss rate was lower than 0.2%) and corresponding retransmissions. The reason for these segment losses is not known. As each TCP segment contains 192 cells, and assuming that a single cell is lost in each lost segment, this corresponds to a cell loss rate of approximately $7*10^{-6}$, which is comparable to the CLR measurements presented in [LP96].

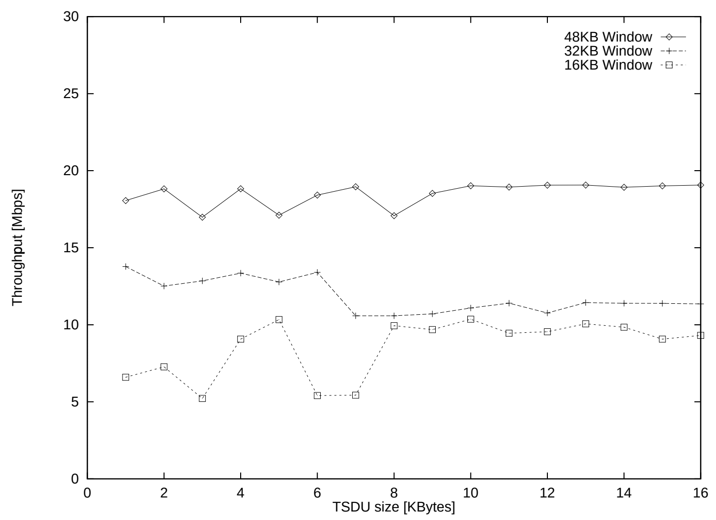

Figure 4.4: TCP throughput with a conforming ATM traffic

[4]By TSDU size, we mean the size of the user data buffer passed to TCP by the ttcp application with the send() system call.

4.3.3 Measurements with an almost conforming traffic

In order to see what happens when the ATM level traffic is not completely conforming, we performed the same throughput measurements with a cell rate of one cell every 15.97 μsec which is the closest non-conforming cell rate achievable with our SBA-200 ATM adapters. With this cell rate, the results were similar to the results presented in figure 4.4.

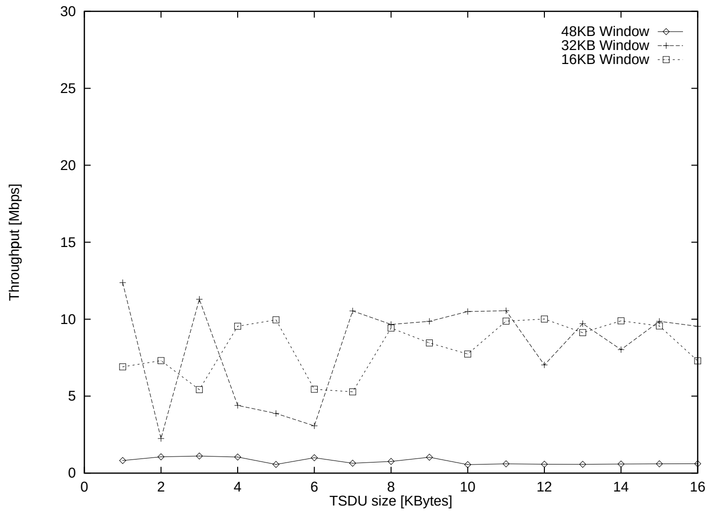

Figure 4.5: TCP throughput with spacing set to one cell every 15.50 μsec

When we tried the next lower value for the spacing (one cell every 15.50 μsec), the TCP throughput almost collapsed. This is shown in figure 4.5. The most affected traffic is the traffic sent with a 48 KBytes window. Figure 4.5 shows that with this window size, the throughput drops below 1 Mbps (compared to 19 Mbps when the spacing was set to one cell every 16.44 μsec). This collapse of the TCP throughput is accompanied by a high segment loss rate, as shown in figure 4.6.

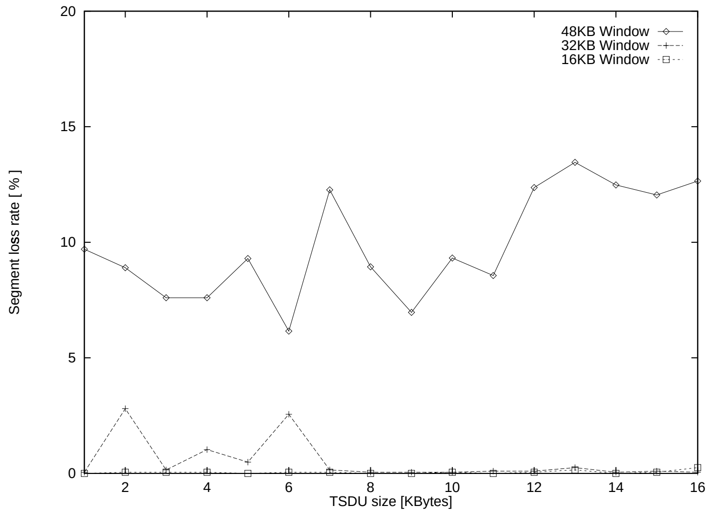

Figure 4.6: Segment loss rate with spacing set to one cell every 15.50 μsec

4.4 Discussion of the first campaign of measurements

4.4.1 Behaviour of TCP with a conforming ATM traffic

During our measurements with a conforming ATM level traffic, the segment loss rate was lower than 0.2%. However, even with such a low segment loss rate, we still measured variations in the TCP throughput. To explain these variations, we gathered packet traces during the measurements. The traces showed that, in SunOS 4.1.3_U1, the lost segments are only retransmitted after the expiration of the retransmission timer. On average, the cost of a single retransmission is 750 msec. This behaviour was caused by a peculiarity of SunOS 4.1.3_U1. While SunOS 4.1.3_U1 will retransmit a segment after 3 duplicate acknowledgements have been received, it does not send an immediate acknowledgement when it receives an out of order segment, and thus the fast retransmit algorithm is only partially implemented in SunOS 4.1.3_U1.

Even with a correct implementation, the fast retransmit algorithm will not necessarily be sufficient to avoid the expiration of the retransmission timer. For random and infrequent single segment losses, this algorithm will only work well if the window contains at least $(d+2)$ Maximum Segment Size (MSS) sized segments, where d is the number of duplicate acknowledgements required to trigger the fast retransmit. If several segments are lost, the window must be even larger.

When a single segment is lost, the worst case for the fast retransmit algorithm occurs when several segments are sent after the connection has been idle for some time. The first segment does not advance the window by at least 2*MSS bytes, and thus the receiver does not send an immediate acknowledgement. If we assume that the second segment is lost, the third segment is out-of-order, and will therefore trigger an immediate acknowledgement. However, as TCP does not distinguish between a standard acknowledgement and an acknowledgement for out-of-order segments, this acknowledgement acknowledges the first segment. At this time, there are already two unacknowledged segments, and d new segments are necessary to generate the d duplicate acknowledgements. Thus, the window must contain at least (d+2) MSS sized segments. With the default value of 9148 bytes for the MSS with TCP over ATM, and a default value of 3 for d, the window should contain at least 45740 bytes. It should be noted that during the data transfer, the slow start and congestion avoidance mechanisms used by TCP may reduce the usable window to values lower than the maximum window size requested at connection establishment. Thus, even if the window is set to $(d+2)$ MSS sized segments when the connection is established, there is no guarantee that the fast retransmit algorithm will always avoid the expiration of the retransmission timer when random losses occur.

4.4.2 Behaviour of TCP with an almost conforming ATM traffic

Our measurements have shown that when TCP is used with the spacing set to one cell every 15.50 μsec (while the peak cell rate enforced by the UPC was one cell every 16.39 μsec), the loss rate increases quickly with the window size. During these measurements, the TCP throughput achieved with a large window was lower than the throughput achieved with a small window.

To explain this unexpected behaviour, we looked at the packet traces gathered during the measurements, and found that when a segment was lost with a 48 KBytes window, it was always the third segment in a burst of at least three segments. However, in a few three segment bursts, the third segment was not lost. From the packet traces, it seemed that bursts of two TCP segments were conforming, while bursts of three TCP segments were not conforming.

This explains why the segment loss rate with a 16 KBytes window is still less than 0.25%, as this window prevents TCP to send bursts of three MSS-sized segments. The difference between the small and the large TSDUs with a 32KBytes window can also be explained by looking at the segments actually sent by TCP. When the TSDUs are smaller than 7 KBytes, TCP may send bursts of 3 segments after a retransmission, and these bursts are declared as non-conforming by the UPC mechanism. When the TSDUs are larger than 7 KBytes, TCP does not send bursts larger than 2 segments, and thus the ATM level traffic is still declared as conforming by the UPC mechanism. This difference in the behaviour of TCP with large and small TSDUs is due to the interactions between the memory management in the socket layer and the transmission of TCP segments in SunOS 4.1.3_U1 [MKK94]. An important point to note is that when the ATM level traffic is not completely conforming, a large window may give a much lower throughput than a smaller window.

The packet traces gathered during the measurements with a 48 KBytes window reveal that the TCP connection is idle for most of the time. Figure 4.7 shows the first 8 seconds of the packet trace corresponding to a 8 MBytes transfer with an almost conforming ATM traffic and a 48 KBytes window. This figure shows that the TCP connection is idle for most of the time and has an almost periodical behaviour. Eight segments are transmitted very quickly (in 4 round trip times), but the sixth segment is lost, and TCP has to wait the expiration of the retransmission timer (two 500 msec ticks - almost one second) to retransmit the lost segment. The larger bursts which occurred around the first and the eighth second contained a few three segment bursts that were declared as conforming by the UPC mechanism.

Even other BSD-based TCP implementations experience similar problems. They correspond to a standard behaviour with the slowstart mechanism. With respect to burst loss, 4.4 BSD [WS95] behaves like SunOS. If the UPC only accepts bursts as long as 2 segments, 4.4 BSD will behave as follows. After a loss, TCP will enter the slow start phase. During this phase, TCP increases its window exponentially, but as soon as the congestion window reaches 3*MSS bytes, a burst of 3 segments is sent, and the third segment is lost. On average, 4.4 BSD will send and acknowledge 7 segments in a period corresponding to 4 round trip times and two ticks (i.e. the minimum value of the retransmission timer), and one segment out of 7 has to be retransmitted. This is because TCP is always in the slow start phase, and does not enter in the congestion avoidance mode. If the UPC is changed so that the largest allowed burst contains 1 segment, 4.4 BSD will transmit on average 3 segments within a period corresponding to two round trip times and two ticks, and half the segments have to be retransmitted. If the non-conformance of the ATM traffic is such that even a single MSS-sized segment is non-conforming, the data transfer is impossible as TCP always tries to use MSS sized segments during a bulk data transfer.

An important point to note from the measurements with an almost conforming traffic is that TCP performs badly when the network limits the maximum burst size to 3 segments or less.

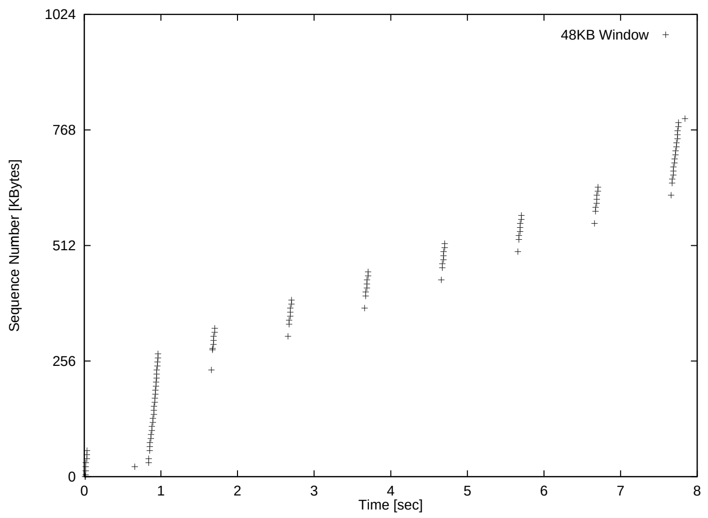

Figure 4.7: First 8 seconds of a trace with the cell spacing set to one cell every 15.50 μsec

4.5 Second campaign of measurements

Our first campaign of measurements has shown that a small variation in the peak cell rate of the ATM level traffic can cause a lot of problems to TCP. To further investigate the behaviour of TCP, we have performed an additional set of measurements. First, we wanted to have a better control over the measurement environment, and especially the parameters of the UPC. For this, we modified the testbed of the University of Liège. A prototype spacer-controller [BSG92] built by CNET was inserted between the output of the Fore ASX-200 ATM switch and the E-3 link to the Belgian switch of the European ATM Pilot (figure 4.8). The spacer-controller performs both traffic shaping and traffic policing. The main advantage of the spacer-controller over a stand-alone UPC, such as the one contained in the Fore ASX-200, is that the cells declared as conforming by the UPC part of the spacer-controller are reshaped at the output of the UPC before being transmitted on the output link. Thus, the ATM cell flow at the output of the spacer-controller has a very low associated CDV, even if its UPC uses a large CDV tolerance.

Unfortunately, when we received the spacer-controller, Electricité de France was not anymore connected to the European ATM Pilot and thus we used the testbed of Telenor Research and Development as the receiver for the second campaign of measurements. The round- trip-time, measured by ping, between the University of Liège and Telenor Research and Development was 31 msec.

For the second campaign of measurements, we requested a 63679 cells per second (i.e. one cell every 15.70 μsec) VP through the ATM Pilot. We used the spacer-controller to enforce a traffic contract of 61000 cells/second (i.e. one cell every 16.39 μsec) with an allowed CDV of 187.5 μsec. As the spacer-controller is also a traffic shaper, we were sure that the cell flow sent in the European ATM Pilot was conforming with the traffic contract enforced inside the network. Thus, if we ignore the rare random cell losses, the only source of cell losses during the second campaign of measurements was the UPC of the spacer-controller.

During the second campaign of measurements, we used a fixed TSDU size of 16 KBytes

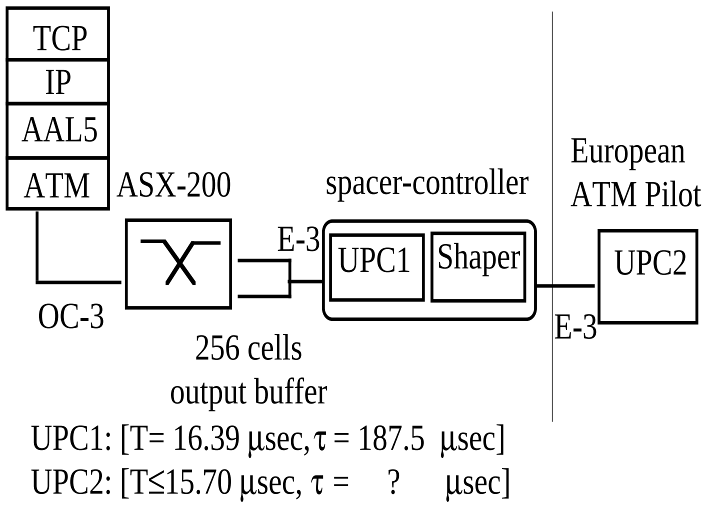

Figure 4.8: Sending side of our environment for the second campaign of measurements

and varied the MTU size (and thus the maximum size of the packets transmitted by TCP) and the traffic shaping on the ATM adapters. We performed our measurements with the MTU set to 9188 bytes (the default value for IP over ATM), 4832 bytes (a value slightly larger than the default value for IP over FDDI), 1500 bytes (the default value for IP over Ethernet) and 552 bytes (which forced TCP to compute an MSS of 512 bytes, which is the default MSS size used by TCP that do not support Path MTU discovery [MD90] when the destination is not in the local subnet). The figures presented in the following sections show the results of individual measurements which are representative of the problems we encountered during our second campaign of measurements.

4.5.1 Measurements with a 48 KBytes window

The measurements with a 48KBytes window are presented in figure 4.9. They show that the variations of the spacing have a lower impact on TCP when it uses small packets than when it uses large packets.

With an MTU size of 9188 bytes, the data transfer was almost impossible with the spacing set to one cell every 15.50 μsec. The packet traces revealed that in this case the longest conforming burst contained only one segment.

With an MTU size of 4832 bytes, there is a moderate throughput drop when the spacing is set to one cell every 15.50 μsec. This throughput drop is caused by a few cells being discarded by the UPC. When the spacing is set to one cell every 15.02 μsec, the data transfer is almost impossible and the packet traces reveal that the longest conforming bursts contained only one segment.

With an MTU size of 1500 bytes, the throughput drop occurs when the spacing was set to one cell every 14.07 μsec. The packet traces revealed that the losses mainly occurred when TCP sent bursts of 8 or 9 segments. When the spacing was set to one cell every 13.60 μsec, the losses mainly occurred when TCP sent bursts of 4 or 5 segments, and the throughput dropped to 250 Kbps. We did not perform measurements with the spacing set to a value lower than one cell every 13.60 μsec.

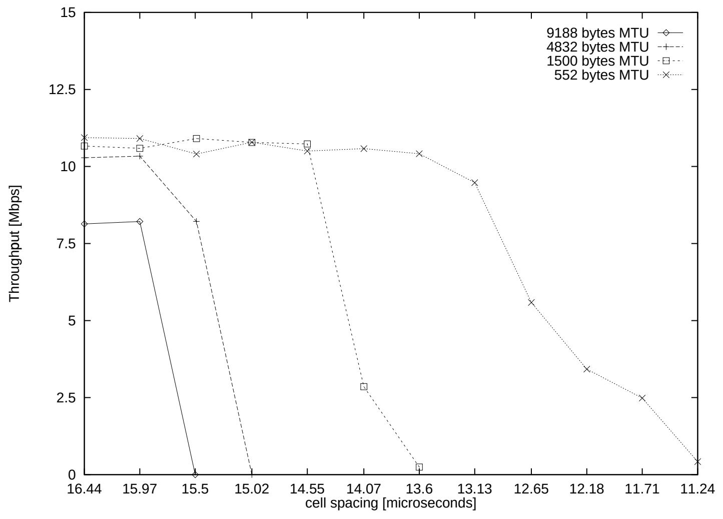

Figure 4.9: Measurements with a 48KBytes window

With an MTU size of 552 bytes, the UPC starts to discard cells and the throughput drops when the spacing is set to one cell every 12.65 μsec. The throughput drop is "smoother" with a small MTU size than with a large MTU size. With the spacing set to one cell every 11.24 μsec, the achieved throughput was approximately 400 Kbps. The packet traces revealed that, with this spacing, bursts of 8 segments were not conforming with the traffic contract.

4.5.2 Measurements with a 16 KBytes window

The measurements corresponding to a window size of 16 KBytes are presented in figure 4.10. With this window size, the throughput drops occur when the spacing is set to one cell every 15.02 μsec with MTU sizes of 9188 and 4832 bytes. With this spacing, the data transfer was almost impossible with an MTU size of 9188 bytes, while it was very slow (less than 100 Kbps) with an MTU size of 4832 bytes.

With an MTU size of 1500 bytes, the throughput dropped to 250 Kbps when the spacing was set to one cell every 13.60 μsec. With an MTU size of 552 bytes, the throughput drops occur when the spacing is set to one cell every 12.18 μsec.

4.6 Discussion of the second campaign of measurements

4.6.1 The maximum throughput

Compared with the first campaign of measurements, TCP only achieved a maximum throughput of 11 Mbps while it achieved 19 Mbps during the first campaign. This difference is, of course, due to the larger value of the round-trip time during the second campaign of measurements. We used a maximum window size of 48 KBytes as the

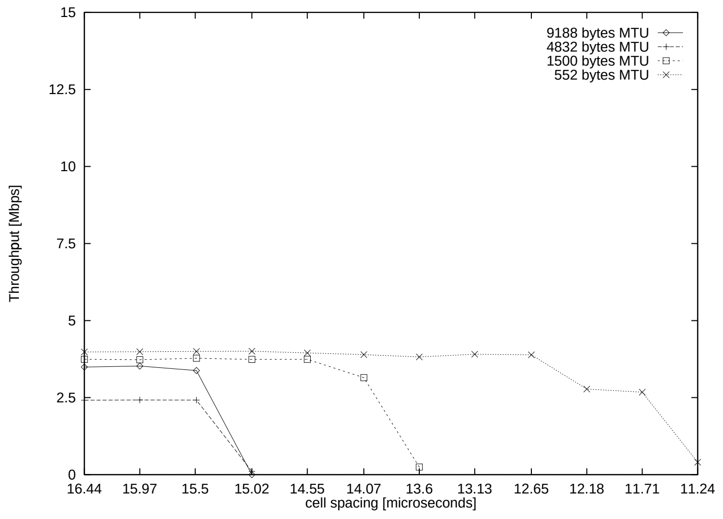

Figure 4.10: Measurements with a 16KBytes window

SunOS 4.1.3_U1 TCP implementation does not support the large windows extensions, and furthermore restricts the window size to 52428 bytes.

Figure 4.9 and figure 4.10 show that, when the cell spacing of the ATM adapter is larger than one cell every 15.02 μsec, a large MTU does not always result in the highest throughput. This can be explained by the Nagle and the Silly Window Syndrome avoidance algorithms. Due to these algorithms, TCP will not send a segment unless it contains MSS bytes or at least half the maximum window size advertised by the receiver. This means that the usable window is actually reduced to an integer number of MSS-sized segments.

Another limitation is due to the delayed acknowledgement strategy. This strategy forces TCP to acknowledge immediately only every second segment. For example, with a maximum window corresponding to 5 MSS sized segments (as with the 9188 bytes MTU and a window of 48 KBytes), when TCP sends 5 segments, the second and the fourth segments will be acknowledged immediately. Thus, within the first round-trip time, TCP has sent 5 segments (segments 1 to 5), but only 4 segments have been acknowledged (segments 1 to 4). During the second round-trip-time, TCP will thus only send 4 segments, and the sixth and eighth segments will be immediately acknowledged. During the second round-trip-time, TCP has sent 4 segments (segments 6 to 9) and received acknowledgements corresponding to 4 segments (segments 5 to 8)... It should however be noted that these reductions of the usable window do not matter when the window size is sufficiently larger than the bandwidth delay product.

4.6.2 The throughput drops

The throughput drops can be explained by looking closely at the behaviour of the SBA-200 ATM adapter. This ATM adapter performs the CRC calculation for AAL5 but contains a limited amount of memory and uses DMA to retrieve the AAL-SDUs from the main memory of the workstation. The traffic shaping is done on a per AAL-PDU basis. Each AAL-PDU is sent at the requested peak cell rate and there is an inter-PDU time between two adjacent AAL-PDUs. This behaviour is captured by the packet train model shown in figure 4.11. The inter-PDU time is probably caused by the preparation of the DMA operation for the next AAL-SDU by the ATM driver. We were not able to measure this inter-PDU time accurately with TCP, but we have measured an inter-PDU time of 52 μsec with 8KBytes long UDP packets (section 4.2.4). As this inter-PDU time is caused by the SBA-200 ATM adapter, and not by the processing in the higher layer protocols, we can assume that the same value also holds for TCP segments. With 552 bytes long UDP packets, this inter-PDU time is reduced to 41 μsec.

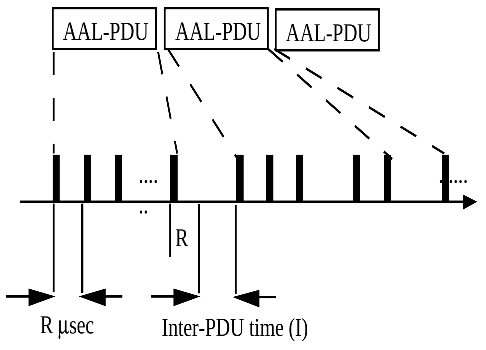

Figure 4.11: Packet train model

From the specification of the GCRA it is easy to see that such a sequence of p packets separated by an inter-PDU time I, with each packet containing c cells sent with the spacing set to one cell every R μsec, will be conforming with the traffic contract [T=T_{UPC} , τ= CDV] if, considering that the UPC is in the idle state upon arrival of the first cells :

$$\forall k = 0...(p-1), \forall j = 0...(c-1) : (k \times c + j) \times T_{UPC} \leq k \times (c \times R + I) + j \times R + CDV \quad (4.3)$$

This packet train model can be used together with equation 4.3 to calculate the length of the longest burst which is conforming with the [T=16.39 μsec, τ=187.5 μsec] traffic contract enforced by the spacer-controller. In table 4.1, we have used an inter-PDU time (I) of 52 μsec for the MTU sizes of 1500 bytes (32 ATM cells), 4832 bytes (101 ATM cells) and 9188 bytes (192 ATM cells) and an inter-PDU time of 41 μsec with an MTU of 552 bytes (12 ATM cells). In table 4.1 , we also take the buffering of the ASX-200 into account when the spacing is set to values lower than one cell every 12.50 μsec (i.e. the cell time on the E-3 interface). In this table, the value 0 means that a single MTU-sized packet is not conforming.

Table 4.1 can be used to explain the throughput drops with an MTU size of 9188 bytes. When the spacing is set to one cell every 15.97 μsec, the largest conforming burst contains 4 segments. TCP is not affected by this limitation, because TCP did not sent bursts containing more than three segments. With the spacing set to one cell every 15.50 μsec and a 48 KBytes window, the model agrees with the measurements. With a 16 KBytes window, due to interactions between the socket layer and the memory management [MG95], TCP sent one 4096 bytes segment followed by one 8192 bytes segment every round-trip-time. This burst of two segments was conforming with the traffic contract.

	MTU Size [Bytes]			
R [μsec per cell]	552	1500	4832	9188
15.97	∞	∞	∞	4
15.50	∞	∞	3	1
15.02	∞	∞	1	0
14.55	∞	20	1	0
14.07	∞	6	0	0
13.60	∞	3	0	0
13.13	∞	2	0	0
12.65	53	2	0	0
12.18	17	1	0	0
11.71	10	1	0	0
11.23	8	1	0	0

Table 4.1: Length of the longest conforming burst (in MTU sized packets)

The measurements with an MTU size of 4832 bytes can also be explained by table 4.1. It should be noted, that due to the inter-PDU time of 52 μsec with our SBA-200 ATM adapters, the traffic sent with the spacing set to one cell every 15.97 μsec was conforming with the traffic contract enforced by the spacer-controller.

The measurements with an MTU size of 1500 bytes almost agree with table 1. With the spacing set to one cell every 14.07 μsec and a 48 KBytes window, the packet traces revealed a maximum burst length of 8 segments, while the model predicts a maximum burst size of 6 segments. This may probably be explained by the fact that the actual transmission of ATM cells is not as regular as the packet train model, especially for long bursts. When an acknowledgement is received while segments are being transmitted, it interrupts the normal transmission of ATM cells, and thus causes a small idle time. When the spacing is set to one cell every 13.60 μsec, the model agrees with the measurements, as the throughput drops heavily.

With an MTU size of 552 bytes and a 48 KBytes window, the measurements indicated that the UPC started to discard cells when the spacing was set to one cell every 12.65 μsec. This explains the throughput drop with this spacing as a 48 KBytes window permits bursts of up to 96 segments with an MTU size of 552 bytes, while the model predicts a maximum burst length of 53 packets. With a 16 KBytes window, the throughput dropped when the spacing was set to one cell every 12.18 μsec. This is also coherent with the model as a 16 KBytes window permits bursts of up to 32 segments with an MTU size of 552 bytes, while the model predicts a maximum burst length of 17 packets.

With an MTU-size of 552 bytes, the throughput drop is "smoother" than with a large MTU size. However, the cost of this "smoother" degradation when the traffic is not entirely conforming is a high CPU utilization. With our workstations, the highest achievable throughput with an MTU-size of 552 bytes is almost 15 Mbps, even in the local area, while the same workstations are able to sustain a 70 Mbps throughput with an MTU size of 9188 bytes. Thus, using a small MTU is not necessarily a wise choice.

4.7 Possible improvements to the TCP implementations

Two modifications to the current TCP implementations could improve the behaviour of TCP in our environment. The first modification concerns the implementation of the timers. In most BSD derived implementations of TCP, including SunOS, the round trip time measurement has a granularity of 500 msec (1 tick), and the retransmission timer has a minimum value of 2 ticks. These default values were chosen to avoid to overload the 1 MIPS workstations that were available when TCP was included in 4.x BSD. At that time, the networks were also much slower than today, and such a high granularity for the round trip time measurements and the retransmission timers did not cause problems. Now, the networks are much faster, and even on the Internet, round trip times lower than 500 msec are very common, and it would be very useful to have a much lower granularity for the round trip time measurements and the retransmission timer. In BSD derived implementations, this may be done by changing the value of the PR_SLOWHZ constant in the kernel.

The second modification concerns the retransmission algorithm. A simple possible change in the fast retransmit algorithm is the value of the duplicate acknowledgement threshold. The default value of 3 for this threshold was chosen to avoid unnecessary retransmissions when the network may reorder packets without loosing them. In an ATM network, the cells are always delivered in sequence, and thus reordering does not occur. In ATM networks, the optimal value for the duplicate acknowledgement threshold would thus be 1. In other networks such as the Internet, where reordering may occur, especially when the route changes, setting the duplicate acknowledgement threshold to 1 is probably too aggressive.

To evaluate the impact of these two modifications, we performed several simulations with STCP[5]. For these simulations, we used the same delays and UPC parameters as for the second campaign of measurements, but we used a 10 milliseconds granularity for the retransmission timer and set the duplicate acknowledgment threshold to 1. Figure 4.12 shows the throughput achieved by this modified TCP implementation.

A comparison between the simulation results (figure 4.12) with a small granularity for the retransmission timer and our measurements (figure 4.9) does not reveal a significant difference. When the ATM level traffic is conforming, both variants of TCP achieve the same throughput. When the ATM traffic is only almost conforming, the TCP throughput drops significantly in both cases. The only difference is that with a 10 milliseconds retransmission timer TCP is able to achieve a slightly higher throughput when the ATM level traffic is non-conforming. However, the throughput achieved in this case is at least an order of magnitude lower than the throughput achieved with a conforming ATM traffic. Simulations performed with a TCP implementation [Mah96] supporting the selective acknowledgements [MMFR96] produced similar results.

In addition to the TCP measurements discussed in the previous sections, we also had the opportunity to perform a detailed performance evaluation of XTPX [MMR93], a transport protocol based on XTP v 3.6 [Eng92] in the European ATM Pilot [KB96].

[5]For these simulations, we slightly modified STCP so that the ATM adapter sent the ATM cells according to the packet train model shown in figure 4.11.

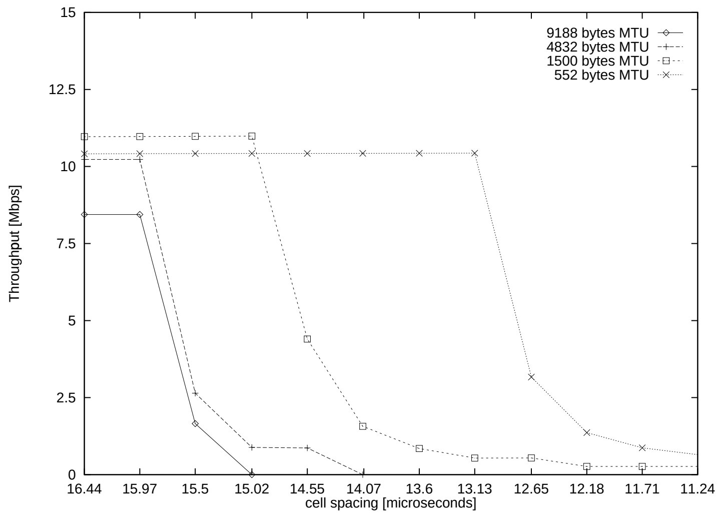

Figure 4.12: Simulations with a 10 milliseconds retransmission timer and a 48 KBytes window

While XTPX provides much better retransmission mechanisms than TCP, it suffered from the same drawbacks as TCP when the ATM traffic was not completely conforming. These measurements confirm the simulations performed with a TCP implementation supporting the selective acknowledgements. We achieved good results with the rate-based flow control mechanism implemented in XTPX, but this mechanism is of limited use in practice since it does not adapt dynamically to the network conditions.

4.8 Possible modifications to the UPC

The modifications discussed in section 4.7 did not significantly improve the performance when the ATM level traffic was only almost conforming. Instead of modifying tailoring TCP to the ATM environment, another possibility would be to adapt the ATM environment to today's TCP implementations. This solution has been used for ATM switches supporting the UBR service category where the switches try to discard tails or entire AAL5-PDUs instead of individual cells when their buffers overflow (e.g. [RF95]).

To allow the endsystems to request the utilization of such packet discarding strategies during overload in ATM switches, the ATM Forum has included a "Frame Discard" option in the UNI 4.0 signaling protocol [For96a]. When this option is selected, the network shall discard entire or tails of AAL5-PDUs instead of individual ATM cells when overload occurs. A similar idea could be used for the UPC. On VCs using "Frame Discard", when the UPC detects a cell as non-conforming, it could discard the tail of the AAL5-PDU (but preserve the last cell of the AAL5-PDU if allowed by the traffic contract). Such a packet discarding scheme can be easily added to a UPC with a cost of one bit per VC. Some switch vendors already support such modified UPCs [Lin96]. Other possible modifications to the UPC are briefly discussed in [SD97].

Simulations performed under the same conditions as the second campaign of measurements did not show a significant improvement in the throughput achieved by TCP with the modified UPC, even with a 10 milliseconds retransmission timer granularity and a TCP implementation supporting the selective acknowledgements. In other environments, the modified UPC might be slightly more useful, but in any case it should not do any harm, and thus could be used by all UPCs when the "Frame Discard" option is selected.

4.9 Management issues

In the previous sections, we have shown that it is very difficult for a transport protocol to adapt its behaviour to a non-conforming ATM level traffic. While the quality of the traffic shapers found on the ATM adapters have improved since we performed our campaigns of measurements, we consider that performance problems due a non-conforming ATM level traffic may occur in production networks, either due to hardware/software problems, or due to limitations of the shapers.

In a production ATM network using SVCs, a shaper could work perfectly for some PCRs and generate a non-conforming traffic with other PCRs. Diagnosing such a problem would not be easy in a real network. In this case, the network operator could rely on the signaling or the management protocols to identify the cause of the problem.

Concerning the signaling protocols, the ATM Forum specifications mention that a UPC may invoke the release of a SVC when the non-conformance of the traffic is unacceptable. However, the UNI signaling protocol does not define any specific error clause that could precisely inform the endsystem. Thus, the RELEASE message due to a non-conformance of the ATM level traffic will probably contain a cause value such as "temporary failure". Such a message would not allow the network manager to detect problems due to non-conforming traffic.

A second possibility is to rely on the Management protocols. The ATM Forum has defined several "management interfaces" [For94b] that are used to exchange management information between separate administrative domains. The M3 interface is concerned with the interactions between the management system of a private network and the management system of a public ATM network. This interface allows the manager of the private network to access some information in the MIBs of the public network. The current MIB defined for the M3 interface [AT94] does not contain any information related to the conformance of the ATM traffic sent across the public UNI. Thus, the current MIB cannot help a network manager to discover cell losses caused by policing. However, a draft addendum to this MIB [LNST97] supports some counters related to the policing. If deployed, these counters will probably be very useful to detect a non-conforming ATM traffic at a UNI.

4.10 Related work

The measurements described in this chapter were performed in 1995 [BDKD95, BKD96a]. Besides us, few researchers have studied the interactions between TCP and the CBR service category. In [DB94], measurements performed with TCP and UDP showing the importance of a traffic shaping capability are presented. Those measurements were performed in an experimental wide area ATM network. The UPC inside the wide area network was disabled, and a 34 Mbps VP was used for the measurements.

The SBA-200 ATM adapters used for these measurements did not yet support traffic shaping, and the two local ATM switches had limited buffers and did not provide a traffic shaping capability. Without the spacer-controller, [DB94] reports that the data transfer was impossible with TCP and the default 9188 bytes MTU size. With UDP, the data transfer was possible provided that UDP sent packets of approximately less than 1 KBytes. The low performance of TCP and UDP in this environment was caused by the lack of sufficient buffering and traffic shaping in the ATM LAN. When the spacer-controller was configured with a PCR of 32 Mbps and a CDVT of 2.788 milliseconds, the TCP became good, and its throughput was only limited by its window size. [DB94] also briefly presents some measurements performed where the spacer-controller enforced different PCRs, but those measurements are not explained in details.

[CHHC95] reports experiments performed with a 1 Mbps VP in the European ATM Pilot. For these measurements, the traffic shaping provided by the release 2.3 of the ATM driver was too bursty and they had to use a Cisco router to perform the traffic shaping.

[MBCM95b] and [MBCM95a] present simulations showing the benefits of traffic shaping when several TCP connections share a bottleneck link with some background traffic.

Since we performed our measurements, the traffic shaping capabilities of the ATM adapters and switches have improved. Several commercially available chipsets contain hardware-based traffic shapers (e.g. [NEC95] or [ATe97]) and ATM switches provide traffic shaping (e.g. [Lin96]). It should however be noted concerning the ATM adapters that while they usually support up to several thousands of VCs, they are usually able to shape a few or a few tens of VCs (e.g. [NEC95] and [ATe97] include 16 shapers while [PS97] supports 8 shapers). However, it should be noted that several recently announced ATM chipsets can be reprogrammed with some firmware like the SBA-200 ATM adapters [Inc96] and thus offer a higher flexibility and more traffic shapers (e.g. [Mot97], [Ins96]).

4.11 Conclusion

In this chapter, we have presented the detailed results of two campaigns of measurements [BKD96a] with TCP over the European ATM Pilot, a wide area ATM network which provided a CBR service. Our main conclusion is that in such a network, it is very important to generate a conforming ATM traffic.

Our measurements have shown that if the ATM traffic is conforming, TCP behaves correctly, and it is usually able to achieve a high utilization of the link. However, TCP relies too much on its retransmission timer, and the high granularity of this timer in most TCP implementations causes a large idle times when random losses occur. When the ATM traffic is only almost conforming, we have shown that the UPC of the ATM network will actually limit the length of the maximum burst. Our measurements have shown that the performance of TCP was highly on the length of this maximum burst. When the maximum burst size allowed by the UPC is smaller than four segments, TCP behaves very badly. When the maximum burst size is larger than three segments, but still smaller than the maximum window size, TCP behaves better but the utilization of the link is still low.

Chapter 5

A critique of ABR

5.1 Introduction

In the previous chapter, we have shown that the CBR service category was able to provide services with a guaranteed bandwidth and that TCP could benefit from this guaranteed bandwidth provided that the ATM level traffic was conforming. However, the main drawback when using the CBR service category to support services with a minimum guaranteed bandwidth is that with this service category the bandwidth reserved for one VC is not available for the other VCs when this VC is not active. This means that a lot of bandwidth can be wasted when the traffic sources are not greedy and as a consequence the service provided by the CBR service category will usually be very expensive. To support TCP/IP with a minimum guaranteed bandwidth, a less expensive service would be very useful.

As mentioned in chapter 3, after the completion of the UNI 3.0 specification by the ATM Forum, many people believed that the service categories defined this specification were not sufficient to efficiently support TCP/IP traffic in ATM networks. To better support data traffic in ATM networks, the ATM Forum has spent several years to define the ABR service category. When the ABR service category was standardized in 1996, many people believed that it would be quickly deployed in ATM networks.

However, more than two years later, the ABR service category hasn't still really started to be deployed. One of the reasons for the delay in the deployment of ABR is its complexity for the ATM adapters and the ATM switches. We analyze the benefits versus cost ratio of the ABR service category to carry TCP/IP traffic in this chapter by summarizing the literature on this subject in section 5.2. We then evaluate in section 5.3 the complexity of the ABR service category by considering the ABR functionality required to connect a corporate network to a wide area public network.

5.2 TCP and the ABR service category

Many papers have studied the performance of TCP with the ABR service category during the last few years. These papers can be divided in two groups depending on the number of TCP connections carried inside each ATM VC. The first group studies the performance of TCP with the ABR service category when a single TCP connection is carried inside each ATM VC. This corresponds to a network where workstations and servers are directly attached to the ATM network. We will use the words "workstation traffic" to identify this type of traffic in the remainder of this thesis. The second group studies the performance of TCP with the ABR service category when several TCP connections are carried inside a single ATM VC. This corresponds to a network where the workstations send traffic through legacy LAN to routers which are attached to the legacy LANs and a backbone ATM network. We will use the words "internetwork traffic" to identify this type of traffic in the remainder of this thesis.

Several of these papers also compare the performance of TCP/IP over ABR with the performance of TCP/IP over UBR. Although UBR does not provide a minimum guaranteed bandwidth which is our focus in this thesis, the comparison between ABR and UBR will allow us to have a closer look at the possible benefits of the ABR service category.

5.2.1 Performance of workstation traffic with ABR

Many papers have studied the performance of workstation traffic with the ABR service category. The papers in this category have looked at the performance of TCP with EFCI-based or explicit-rate based ABR switches in LAN, WAN and even satellite environments. However, to our knowledge, most of the papers in this category have considered that ABR was only providing a best-effort service, i.e. all the ATM VCs used a zero MCR. However, the findings of several of these papers are still worth being discussed.

Several researchers have compared the performance of TCP over UBR with the performance of TCP over binary-mode ABR. In this case, the major problem to be solved is the selection of appropriate values for the RIF and RDF parameters of the ABR traffic contract. The simulations reported in [FL97, HOMM97, IKK97] show that it is very difficult to correctly tune these parameters and although binary-mode ABR can be supported rather easily in ATM switches, it does not appear to be a suitable solution for real networks. The comparison between binary-mode ABR and UBR shows that from an efficiency point of view there is no significant difference between binary-mode ABR and UBR although ABR provides a better fairness than UBR in general. [FL97] notes that "ABR does not provide any significant performance improvement over the less expensive UBR-based services". Other researchers reached similar conclusions.

The comparison between TCP over UBR and TCP over ABR has also been carried out by several researchers [LST+96b, AAA98, ST96] with respect to explicit-rate ABR. In this case, the setting of the parameters of the ABR traffic contract is less an issue since most explicit-rate switch algorithms try to avoid relying on the RIF and RDF parameters. The performance of TCP over explicit-rate mode ABR appears to be better than over binary-mode ABR. As with binary-mode ABR, the comparison between UBR with EPD and explicit-rate ABR shows that TCP achieves a similar efficiency with both service categories but that ABR provides a better fairness than UBR. However, [OA97] shows that there are some performance problems when ABR is used to carry low bandwidth TCP traffic. Besides this and since ABR provides a loss-free service, the performance of TCP over ABR is mainly driven by ABR and some researchers have proposed to introduce closer interactions between the window-based congestion control mechanism used by TCP and the ABR rate-based mechanism by modifying TCP to change its congestion window as a function of the feedback (ER) received by the ATM layer. We do not expect that such modified TCP implementations will be widely deployed.

Several papers have also studied the performance of TCP/IP over ABR with more bursty TCP/IP traffic. These papers [HS98] [VKJ$^+$97] model bursty www applications running over TCP. In [HS98], the performance of explicit rate ABR is compared with the Quantum Flow Control (QFC) credit based congestion control mechanism [All97] which is close to the CT service category. This paper uses the ERICA algorithm [KJF$^+$97] for the ABR switches and a simple credit allocation algorithm in the QFC switches. It only considers a best-effort service since QFC does not support a minimum guaranteed bandwidth. The simulations discussed in this paper show that ERICA and QFC provide similar performance when the bottleneck links are lightly loaded, but that the performance of QFC is better than the performance of ERICA when both the load increases and there is some high-priority background VBR traffic. [VKJ$^+$98] also discusses the performance of TCP over ABR with bursty www traffic. This study considers a simple network scenario with a single bottleneck link. It focuses entirely on the ERICA+ switch algorithm [KJF$^+$97] for the ABR switches and shows that the buffer requirements in the bottleneck switch are of the same order of magnitude as the bandwidth-delay product and only slightly depend on the number of www clients and servers. [KJF$^+$98] provides similar buffer requirements when considering infinite TCP sources. Unfortunately, neither [HS98] nor [VKJ$^+$97] have UBR as a reference configuration to evaluate the real benefits of ABR with www traffic.

Due to the lack of commercial ABR adapters, all the papers discussed above have relied on simulations to evaluate the performance of TCP over ABR. The only measurement study that we are aware of are the experiments described in [NH98]. This paper describes a few measurements performed in a LAN environment with two workstations, a VBR background source and a simple explicit-rate based ABR switch with small buffers. These measurements compare the performance of ABR and UBR and show that ABR requires less buffer than UBR to achieve the same TCP performance. Through an appropriate parameter tuning, they also show that TCP over ABR can outperform TCP over UBR. However, the performance appears to be sensitive to the selection of the parameters of the ABR traffic contract (mainly the RIF). It is unfortunately difficult to extrapolate the results discussed in [NH98] in larger networks.

5.2.2 Performance of internetwork traffic with ABR

The utilization of ABR to efficiently support internetwork traffic has also been studied by several researchers. One of the first papers dealing with this problem is probably [JY97]. This paper shows that when the amount of buffering in the routers is relatively small, then UBR with EPD provides a similar performance as explicit-rate mode ABR. It then shows that the performance of ABR can be improved provided that large buffers are used in the routers attached to the ATM backbone. However, [JY97] also notes that providing large buffers in these routers may not be a very cost-effective solution. Thus ABR requires large buffers in the routers connected to the ATM network, while UBR requires large buffers in the ATM switches but not in the routers.

[OA97] compares the performance of TCP over ABR (explicit-rate mode) with TCP over UBR in an ISP environment. When routers are used to aggregate traffic from several TCP connections, the simulations discussed in this paper show that ABR does not provide an advantage over UBR from an efficiency point of view, although ABR provides a better fairness than UBR. As a conclusion, [OA97] notes that "UBR and ... ABR are both quite good and there is no clear reason to prefer one over the other".

In [MB97], a detailed comparison of the performance of ABR and UBR to support internetwork traffic is carried out through simulations. This paper compares ABR with the ERICA switch algorithm with plain UBR and UBR with EPD. The simulations discussed in this paper show that "from the point of view of overall TCP goodput for LAN interconnection, UBR offers better performance than ABR provided that there is adequate buffering, 8000 cells being sufficient". According to [MB97], the main reasons for the lower performance of ABR compared to UBR are the RM cell overhead and the increasing delay in the routers with ABR.

In order to improve the performance of ABR with internetwork traffic, several researchers [SDK98, NS97] have proposed to introduce special functions in the routers attached to the ATM backbones to "translate" the ABR feedback for the TCP sources. For example, [NS97] proposes to control the rate at which the TCP acknowledgements are returned to the TCP sources in order to reflect the current level of congestion in the ABR-based ATM backbone. Other researchers [SDK98] have proposed to modify the window size of the TCP acknowledgments in transit through the router. However, these mechanisms are rather ad hoc solutions and we do not expect them to become widely used in real networks.

[RAGG98] proposes to use ABR VPs inside an ATM backbone to carry a large number of UBR VCs and compares the performance of these ABR VPs with UBR VPs. The simulations discussed in this paper show that the ABR VPs provide both a higher efficiency and a better fairness than the UBR VPs. [RAGG98] concludes that such ABR VPs are useful to carry a large amount of best effort traffic in backbone networks.

5.3 Requirements for ABR deployment

The literature discussed in the previous sections has shown that from a performance point of view, the ABR service category does not necessarily provide the expected improvement over the much simpler UBR service category if we consider only TCP/IP based best-effort traffic.

However, even if from a performance point of view ABR does not provide a substantial benefit over UBR to support internetwork traffic, one of the benefits of ABR for a network operator is that it allows the operator to control the allocation of the bandwidth to the different sources attached to the network. This type of control is not easily achieved with the UBR service category[1].

While the control provided by the ABR service category appears interesting, it does not come without a cost. To understand the cost of ABR, let us consider an ATM-based corporate network willing to utilize a public ABR service. In this case the complexity of ABR will be mainly located in the following equipments :

- ATM adapters

[1]We note however that, with the GFR service category, the network operator has at least some control on the distribution of the available bandwidth since each source should at least receive its MCR.

- Private ATM switches
- Public ATM switches

The ATM adapters, either on the workstations or on the bridges or routers, which need to support the ABR service category will be much more complex than the current ATM adapters. For the ABR adapters, the major complexity introduced by ABR is the processing of the RM cells and the requirement to modify the rate of all the active VCs in response to the feedback received from the network. Due to the complexity of the calculations required by the ABR source and destination endsystem behaviours, a dedicated CPU or DSP will probably be required in each ATM adapter. The second requirement is the dynamic shaper. The ABR adapter must be able to change dynamically the rate of all its active VCs. For a server, this means that the adapter may need to dynamically shape several hundreds or thousands of VCs[2].

To evaluate the complexity of the private and public ATM switches, we first have to look at the requirements for ABR policing.

5.3.1 The need for ABR policing

For the CBR and VBR service categories, the policing units are used to protect the network and all the established VCs from misbehaving users. Without a UPC and with these two service categories, a misbehaving user could completely jeopardize the QoS of the normal VCs. Since the ABR service category includes a feedback mechanism, it is interesting to see how an ABR network can adapt to a misbehaving user.

As an exercise, we consider a simple scenario where ABR is used to provide a best-effort LAN interconnection service. We consider that five different companies use a public ABR service to interconnect their respective LANs. Each LAN contains twenty TCP sources or destinations and is connected through a router with an ABR adapter to the public ATM network. The TCP sources reside on the LANs on the left hand side of figure 5.1 and send traffic to the destinations connected to a companion LAN on the right hand side. We model the public ATM network with two switches and a single bottleneck link. We assume that all the ATM links have a bandwidth of 34 Mbps. The delay on the link between the ATM switches is set to 10 milliseconds, while the delays on the links between the ATM switches and the routers is set to 2.5 milliseconds. The switches have per-port output buffers of 16.000 cells while the routers are tail-drop routers with 700 KBytes buffers. Two companion routers (e.g. R1 and R1*) are always linked with one ABR VC. The main parameters of the ABR traffic contracts of these VCs are summarized in table 5.1. We used a value of 1 for RIF to ensure that the routers are able to respond immediately to the explicit-rate feedback sent by the network and we set the ICR and the MCR to 1000 cells per second (424 Kbps) to ensure that each router is always able to send some traffic. The sum of the MCRs of the routers was only $\frac{1}{16}$ of the bottleneck link. The value of the RDF is irrelevant since the ATM switches we use for the simulations do not set the CI bit.

[2]At this point, it is worth to recall that, as already mentioned in section 4.10, most of the currently available ATM adapters are only able to shape a few CBR VCs. Shaping a large number of concurrent VCs, even at a fixed rate is not a common feature in today's ATM adapters.

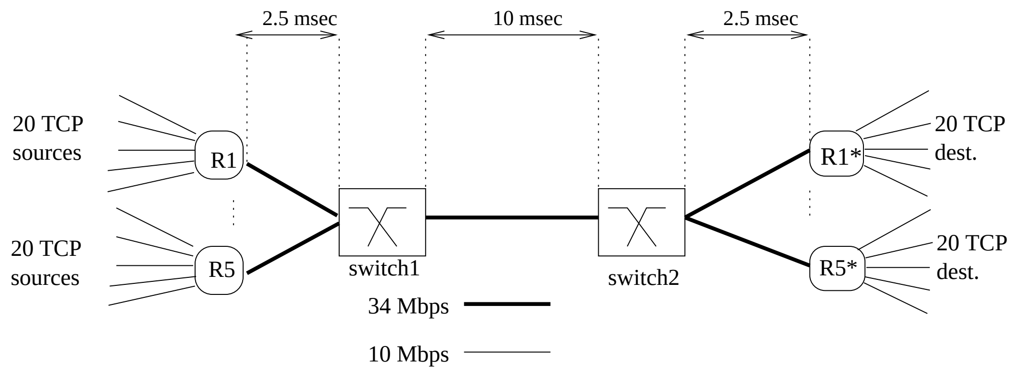

Figure 5.1: The five routers ABR scenario

Table 5.1: Characteristics of the ABR VCs

Parameter	Value
PCR	34 Mbps
MCR	424 Kbps
ICR	424 Kbps
RIF	1
RDF	N/A

The ATM switches use the ERICA+ algorithm [KJF+97] to calculate the feedback to be sent in the backward RM cells. The ERICA+ algorithm is an explicit rate switch algorithm which modifies the ER field of the backward RM cells, but does not modify the CI or NI fields of the RM cells. Like most ABR switch algorithms, ERICA+ assumes that the cells from all the ABR VCs directed to the same output link are multiplexed in a single FIFO buffer. A detailed description of ERICA+ is outside the scope of this thesis, but may be found in [KJF+97]. For this simulation study, we use the parameters recommended by the designers of ERICA+ for a WAN network (table 5.2).

Table 5.2: Parameters for the ERICA+ algorithm

Parameter	Value
Measurement window	2 milliseconds
α for load averaging	0.0625
δ decay factor for VC contribution averaging	0.25
Queue control option	Enabled (a=1.15, b=1.05)
Empty time	1 millisecond
Queue Drain Limit Factor	0.5

Instead of following exactly the source endsystem rules [For96c], we consider that there is one misbehaving router which always transmits at a higher rate than the ER conveyed in the backward RM cells sent by the network. We model this misbehaving router by assuming as in [WPM97] that it computes its transmission rate as shown in equation 5.1 where M_{factor} is a number larger than or equal to one.

$$rate = min(M_{factor} \times min(rate, ER), PCR) \tag{5.1}$$

As a verification, we first considered the case where all the routers follow the ABR source and destination endsystem rules correctly (i.e. $M_{factor} = 1$ for the misbehaving router). In this case, all the routers attached to the ABR network receive the same feedback from the network, and transmit at the same rate (about 6.8 Mbps). The total TCP goodput of the twenty TCP sources attached to each router is equal to 5.6 Mbps. There are some packet losses in the routers, but ERICA+ is able to perfectly control the queue length on the ATM switches and no packets are lost in these switches. Thus ERICA+ behaves correctly in this case.

To evaluate the influence of a misbehaving router, we set the M_{factor} of one of the routers to 1.5. This means that the misbehaving router will always try to transmit at a rate which is 50% larger than the ER feedback received from the network. The simulations performed with this misbehaving router show that by using this strategy this router is able to steal bandwidth from the well-behaving routers. Figure 5.2 compares the transmission rate of the misbehaving router with the transmission rate of a normal router. This figure shows clearly that the misbehaving router always transmit at a higher rate than the normal routers. After about 30 seconds, the rates of the misbehaving and the normal routers stabilizes at around 16 Mbps for the misbehaving router and 4 Mbps for the normal routers. Even with such a misbehaving router, the ERICA+ algorithm was able to correctly control the buffer occupancy on the ATM switches and no cells were lost in these switches.

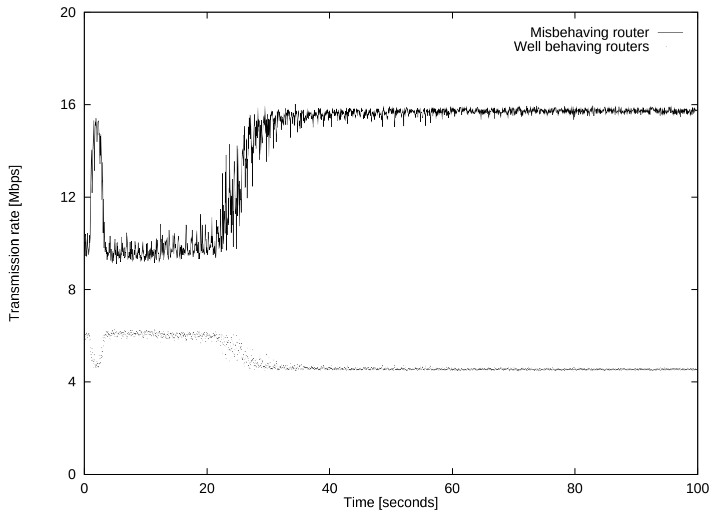

Figure 5.2: Impact of the misbehaving router

This simple simulation shows clearly that, at least with the ERICA+ switch algorithm, a source can receive a large fraction of the available bandwidth when it does not follow correctly the source endsystem rules. Furthermore, there is a kind of "unwanted cooperation" between the misbehaving source and the ERICA+ algorithm which forces the normal sources to transmit at a much lower rate than their fair share. Thus, ERICA+ favors the misbehaving router at the detriment of the well behaving routers.

We believe that most of the switches designed to work with all the cells from all the ABR VCs in a single FIFO queue also suffer from problems when confronted with such misbehaving sources [3]. For some of these algorithms, the problem might be that cell loss occurs inside the ATM networks. This is typically the case for switch algorithms which do not take the queue length into account (e.g. ERICA). Others may behave like ERICA+, i.e. the switch provides a loss-free service, but the misbehaving source is favored over the well behaving sources.

This simulation clearly shows that some form of protection from misbehaving sources will be required in public networks which support the ABR service category. This protection can either be implemented as an ABR policing unit at the ingress of the public network or by using ATM switches with Virtual Source/Virtual Destination (VS/VD) capabilities at the ingress of the public network.

Let us first discuss the impact of the utilization of ABR policing at the ingress of the public network. As already mentioned in section 3.5.3, the ATM Forum and the ITU-T have defined conformance definitions [For96c] [ITU97] for the ABR service category. These conformance definitions should be the basis for the development of ABR policing units. It should however be noted that these conformance definitions are much looser than the conformance definitions used for the CBR and VBR service categories. The conformance definition for these two service categories is a precise algorithmic description of the traffic contract. It can thus be used to identify whether a source obeys completely to the negotiated traffic contract. With the ABR service category, the conformance definitions proposed in [For96c] and [ITU97] are only an approximation of the source and destination endsystems rules defined in [For96c]. The ABR conformance definitions are currently only capable of verifying whether the endsystems respond correctly to the ER imposed by the network in the backward RM cells, but not the CI and NI fields. The ABR conformance definition is neither capable of verifying whether the endsystems applies the RIF, RDF and other parameters of its traffic contract. This implies that, even when ABR policing is used at the ingress of the public network, the public ATM switches can only use explicit-rate switch algorithms which do not rely on the CI and NI fields of the RM cells.

If ABR policing is deployed at the ingress of the public network, it can be expected that it will be used with relatively low τ_1, τ_2 and τ_3 tolerances in order to efficiently protect the network from misbehaving users. However, these low tolerance values will in practice impose a reshaping of the ABR traffic at the egress of the corporate network. To perform this shaping, the ATM switches of the corporate network attached to the public network will have to implement the VS/VD capabilities.

[3]Switch algorithms based on the Virtual Source/Virtual Destination (VS/VD) principle [KJJ+98] [RAGG98] [Cis98] [GCJ+98] or relying on per-VC queuing and scheduling such as the algorithm proposed recently in [CW97a] should be able to reduce the impact of a misbehaving source since they isolate the traffic from the different VCs. However, these switches are much more complex than the FIFO-based switches which are usually considered to support the ABR service category.

The other option for a public network operator would be to use ATM switches with VS/VD at the boundaries between administrative domains. Since these switches break the ABR feedback loops, they should be capable of correctly protecting the network and the well-behaving sources from misbehaving sources, but they are much more complex than the simple FIFO-based switches.

5.4 Conclusion

In this chapter, we have first summarized the literature on TCP/IP with the ABR service category and shown that the benefits of ABR over UBR were questionable. Thus, from a pure performance point of view, the benefits of ABR when carrying TCP/IP traffic are not as high as initially expected.

We have then studied the complexity of ABR and shown that to deploy ABR in public networks, either private or public ATM switches with VS/VD capabilities were required to cope with policing or misbehaving users. Since these switches need to implement the complete ABR source and destination endsystem behaviours for potentially a large number of VCs, it can be expected that they will be much more complex than ATM switches which only support the classical service categories (i.e. CBR, VBR and UBR).

While ABR will probably be deployed in some corporate networks, for instance to support non-TCP/IP traffic and inside public ATM networks to control the best-effort traffic at VP-level, we do not believe that ABR will be widely deployed and thus the dream of having end-to-end ABR VCs [Rob97] in an ATM-based Internet will probably never become a reality.

Chapter 6

TCP and the VBR service category

6.1 Introduction

In the previous chapter, we have shown that the complexity of the ABR service category was a big burden towards its deployment, especially in public networks. Due to this complexity, there is few hope that ABR will become widely available. To efficiently support TCP/IP services with a minimum guaranteed bandwidth, we have to look at other service categories. We have already shown the limitations of CBR service category in chapter 4. In this chapter, we evaluate whether the VBR service category can be used to efficiently support services with a minimum guaranteed bandwidth.

This chapter is structured as follows. We first briefly describe in section 6.2 the most common implementation to support the VBR service category in ATM switches. Then we discuss in section 6.3 measurements carried in an ATM LAN to evaluate the suitability of the VBR service category for workstation traffic. We then rely on simulations to perform a similar analysis with internetwork traffic in section 6.4. Finally we discuss some related work in section 6.5.

6.2 VBR switch implementations

A typical first generation ATM switch which only supported the CBR service category contained a few tens or perhaps a few hundred cells of buffers and relied solely on the UPC at the ingress of the network and on its Connection Admission Control (CAC) algorithm to ensure a loss free operation. These small buffers are not sufficient to efficiently support (i.e. with a high statistical multiplexing gain) the bursty VBR traffic in ATM switches. Thus, most switches which support the VBR service category use buffers containing several thousand or several tens of thousand cells.

While the CBR service category is now completely transparent to the CLP bit, the VBR service category supports two possible utilizations of this bit. The first utilization is similar to the CBR service category, i.e. the CLP bit is completely transparent for the network and the QoS commitments of the network in terms of cell losses are applicable for the CLP=0+1 cell flow. This corresponds to the VBR.1 conformance definition. To fulfill the QoS commitments, the switches should not discard any cells from VBR.1 VCs.

The second utilization corresponds to the VBR.2 and VBR.3 conformance definitions.

With these conformance definitions, the QoS commitments of the network in terms of cell losses apply only to the CLP=0 cell flow. For the ATM switches inside the network, there is no real difference between the two conformance definitions. In both cases, the switch should not discard CLP=0 cells, but they can discard CLP=1 cells when congestion occurs.

To provide a low CLR for CLP=0 cells, most ATM switches implement the Partial Buffer Sharing (PBS) [KHBG91] buffer acceptance algorithm. This algorithm (figure 6.1) keeps track of the occupancy of the buffer used for the VBR cells and accepts new CLP=1 cells provided that the buffer occupancy is below a predefined threshold ($PBS_{threshold}$). The value of this threshold is determined by the CAC algorithm and depends on the burstiness of the established VBR VCs.

```
Cell arrival on VC[i] :


switch(conf_def(VC[i]))
    {
        case VBR.1:
            if($B_{Occupancy} = B_{Size}$)
                discard(cell);
            else
                accept(cell);
            break;
        case VBR.2:
        case VBR.3:
            if($B_{Occupancy} = B_{Size}$)
                discard(cell);
            else
                if($B_{Occupancy} < PBS_{threshold}$) OR (IsCLP0(cell))
                    accept(cell);
                else
                    discard(cell);
            break;
    }
```

Figure 6.1: Sample buffer acceptance algorithm for a VBR switch

Other buffer acceptance algorithms have been proposed in the literature. Another largely discussed solution to support the VBR service category is the pushout buffer acceptance algorithm [KHBG91] where an arriving cell may replace a cell which is already in the buffer. Although simulations and analytical studies have shown that pushout provides a better performance than PBS, pushout is difficult to implement at high speed and most switches rely only on PBS.

In this chapter, we will not consider the VBR.1 and VBR.2 conformance definitions. The main drawback of the VBR.1 conformance definition is that the endsystems are not allowed to transmit at a higher rate than their reserved bandwidth (SCR,MBS), even when the network is not congested. Thus, although it allows a slightly more bursty traffic, the

VBR.1 conformance definition will provide a similar service as the CBR service category provided that the endsystems fulfill their traffic contract. We do not consider the VBR.2 conformance definition because we are not aware of ATM adapters which are able to tag the cells which are transmitted in excess of the reserved bandwidth (SCR,MBS) to respect the VBR.2 conformance definition.

6.3 Workstation traffic over VBR

As a first evaluation of the suitability of the VBR service category to efficiently support workstation traffic, we carried several measurements in a simple ATM LAN during the first half of 1996. Although these measurements are limited by the available equipments, they will allow us to gain a good understanding of the interactions between TCP and the VBR service category.

6.3.1 Measurement environment

Our measurement environment (figure 6.2) is based on the measurement environment used in chapter 4. For the VBR measurements, it consisted of two Sparc 10 class workstations and one ASX-200BX ATM switch from Fore Systems. The two workstations were running SunOS 4.1.3_U1 and were connected to the ASX-200BX with 155 Mbps links. The sending workstation used a Fore SBA-200 ATM adapter, while the receiving workstation used a more recent Fore SBA-200E ATM adapter. They both used release 4.0 of the ATM driver. We used the same SunOS 4.1.3_U1 TCP implementation as for the measurements discussed in chapter 4, but we fixed the fast retransmit bug for the measurements discussed in this chapter.

Due to the limited size of our ATM LAN, we could not multiplex the VBR VCs from several workstations on a single bottleneck link. Instead, we considered a worst case, but not unrealistic, scenario for the VBR service category. With the PBS-based switch implementations, the value of the $PBS_{threshold}$ is usually chosen by the CAC algorithm in order to meet the QoS commitments for the established VCs. If the traffic contracts of the established VCs allow a large burstiness, then the CAC algorithm will have to reserve most, if not all of the buffer, for the CLP=0 cells. In this case, the value of the $PBS_{threshold}$ will be close or equal to zero. This is the case we consider for our measurements. However, instead of changing the $PBS_{threshold}$ of our switch manually, we configured its UPC to discard the non-conforming cells. This setting results in the same behaviour as a UPC which tags the non-conforming cells according to the VBR.3 conformance definition and a PBS-based switch which discards all the CLP=1 cells since in both cases the CLP=1 cells are discarded.

For our measurements, we established one bi-directional PVC between the two workstations. There was no other ATM traffic through the switch during our measurements, and thus there was no congestion inside the switch. We choose a Peak Cell Rate of 155 Mbps (i.e. one cell every 2.726 μsec which is equal to the physical line rate), no specified allowed Cell Delay Variation for the Peak Cell Rate, a Sustainable Cell Rate of 10 Mbps (i.e. one cell every 42.4 μsec) for the VBR traffic contract. We then varied the Maximum Burst Size during the measurements.

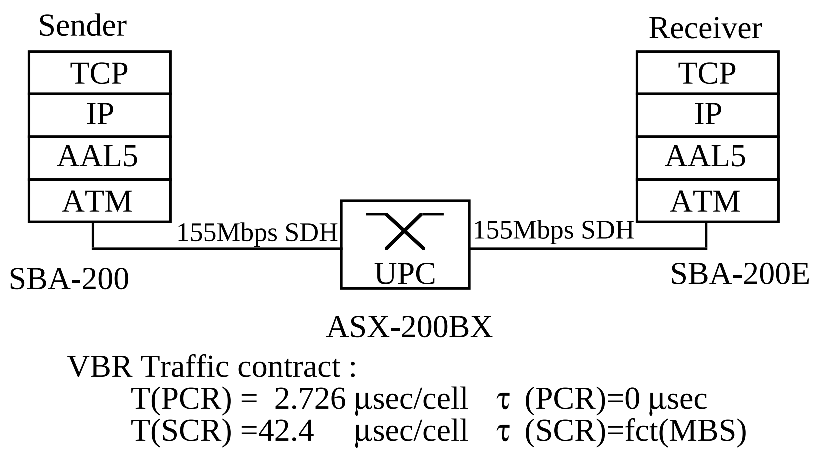

Figure 6.2: Measurement environment

Our measurements were performed in the environment shown in figure 6.2 with the standard `ttcp` [SM84] measurement tool. Each measurement was reproduced several times, and consisted in the memory-to-memory transfer of 4 MBytes of data in 16 KBytes long buffers. We enabled the TCP_DEBUG option for all the TCP connections to gather some packet traces. The receive and transmit window sizes were always set to 49152 bytes on both the sender and the receiver. By using the same transmit and receive windows on both workstations, we avoid the deadlock problems with SunOS 4.1.3_U1 discussed in [MG95]. The round-trip-time measured by ping with small packets between the sending and the receiving workstation is about 1800 μsec.

6.3.2 First Measurements

For our first measurements, we varied the MTU size used by IP over ATM. The selected MTU sizes correspond to the most commonly used MTU sizes for IP over ATM[1]. We also varied the Maximum Burst Size enforced by the UPC.

We choose 1000 cells as the minimum value for the MBS since we have already discussed in chapter 4 how TCP behaves when a UPC enforces a MBS corresponding to only a few packets. We choose 4000 cells as the maximum value for the MBS since our ASX-200BX (whose CAC algorithm is based on [GAN91]) could not establish (and associate QoS guarantees to) one ATM VC with a MBS much larger than this value.

The results of our first measurements (figure 6.3) were disappointing. We established one 10 Mbps VBR VC through the switch to benefit from QoS guarantees, and the throughput achieved by TCP with the larger MTU sizes is below 1 Mbps. The throughput achieved by TCP with an MTU size of 552 bytes is slightly higher, but still far from satisfactory. This is obviously a too small utilization of the resources reserved for the 10 Mbps VBR VC.

A look at the packet traces gathered during the measurements revealed that TCP sent

[1]552 bytes corresponds to the default segment size used by TCP in the wide area, 1500 bytes corresponds to an Emulated Ethernet [For95], 4832 bytes to an Emulated TokenRing[For95], and 9188 bytes is the default MTU size for IP over ATM[Atk94]

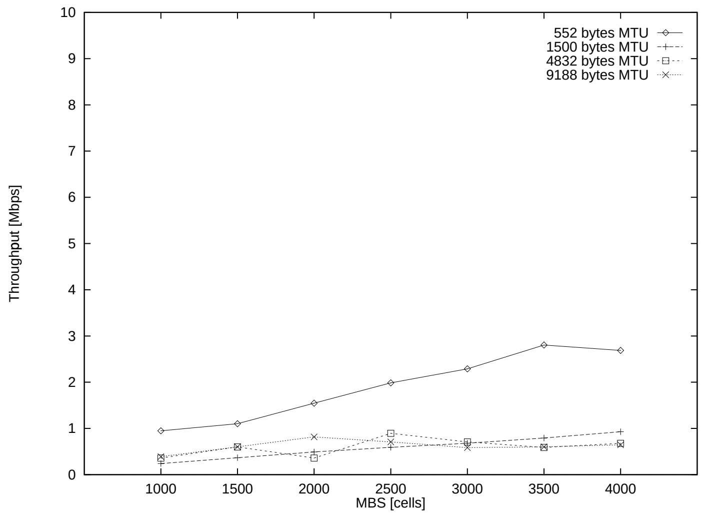

Figure 6.3: First Measurements with TCP

bursts of packets separated by an idle time of roughly one second. The TCP statistics reported by `netstat -s` showed a large number of retransmitted packets.

The packet losses which caused this large number of retransmissions are due to non-conforming cells being discarded by the UPC. This can be explained by looking at the behaviour of our ATM adapter and at the GCRA used by the UPC. The large idle times, which caused most of the throughput drops, are due to the implementation of TCP's retransmission timer in SunOS and BSD-derived implementations and will be discussed later.

Explanation for the packet losses

To explain the packet losses, we can consider the packet train model shown in figure 6.4. This model is close to the behaviour of our ATM adapters. When a TCP packet is sent by the SunOS kernel, it is encapsulated inside one IP packet and passed on to the ATM driver. The driver enqueues this AAL-SDU and then wakes up the firmware of the ATM adapter. The AAL5 segmentation is performed inside the ATM adapter before the transmission of the corresponding ATM cells on the output link. In our environment, the transmission of one AAL-PDU by the ATM adapter took less time than the generation of the TCP/IP packet and the ATM driver processing. Thus, two consecutive AAL-PDUs were always separated by some idle time on the output link.

The ATM traffic was always conformant to the PCR, as we set the PCR of the traffic contract to the PCR of the 155 Mbps link. Thus, all the cells discarded by the UPC were discarded due to a non-conformance with the SCR and MBS parts of the VBR traffic contract. From the specification of the GCRA and the packet train model, it can be shown that a sequence of p AAL-PDUs separated by an inter-PDU time I_p, with each packet containing c cells, will be compliant with the traffic contract enforced by the GCRA if, considering that the UPC is in the idle state upon arrival of the first cells :

$$\forall k = 0...(p-1), \forall j = 0...(c-1) :$$

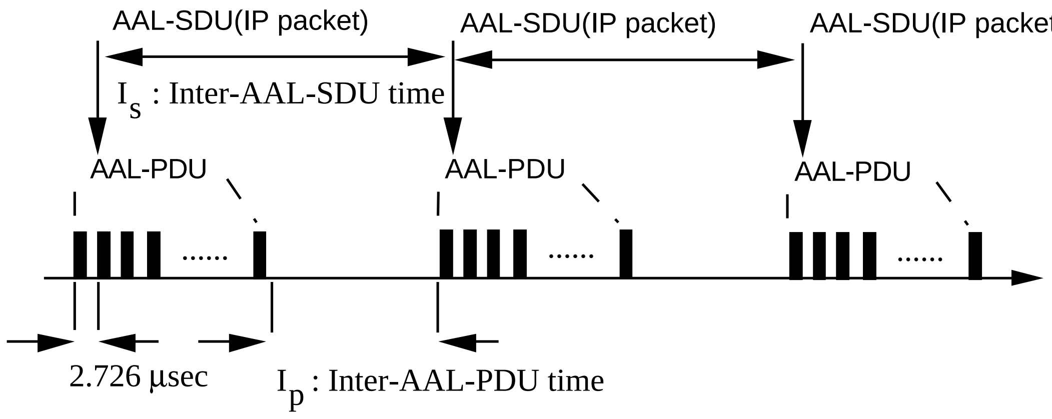

Figure 6.4: Packet Train Model

$$(k \times c + j) \times T_{SCR} \leq k \times (c \times 2.726 + I_p) + j \times 2.726 + \tau_{SCR} \tag{6.1}$$

In an ATM LAN, the behaviour of TCP is very close to this packet train model. We were not able to measure the Inter-AAL-PDU time directly, as this would have had required an ATM analyzer which was not available where we performed our measurements. Instead we measured the Inter-AAL-SDU time with a modified (but older - release 2.2.9) Fore ATM driver. This modified driver [BKD96a] is able to log in a kernel table a timestamp and some related information for each AAL-SDU sent or received by the driver. When we used TCP with this modified driver, the timestamps showed that there were some variations in the Inter-AAL-SDU times. These variations occurred, e.g. when TCP had to process one received acknowledgment or fetch some data from the user process' buffer. However, 600 μsec and 900 μsec appeared as reasonable estimations for the Inter-AAL-SDU times with respectively the 4832 and 9188 bytes MTU sizes . This corresponds to respectively 324 and 376 μsec for I_p, and to raw throughputs of respectively 71 Mbps and 90 Mbps.

The packet traces have shown that TCP transmitted burst of packets separated by an idle time of roughly one second. Figure 6.5 compares the maximum burst length (in packets) predicted by the packet train model with the average burst size found during the measurements. The model seems to be a good approximation of the measured values. The main reason for the difference between the measurements and the model is the slow-start mechanism used by TCP. When TCP starts to transmit after the expiration of its retransmission timer, it sends one packet and waits for the corresponding acknowledgement before sending the second one. In this case, the time to wait for the acknowledgement is larger than I_p. When the acknowledgement for the first packet is received, TCP sends two packets and waits for the corresponding acknowledgements... Since the round-trip-time is very small in our LAN, the behaviour of TCP becomes aligned with the model when the congestion window reaches between 2 and 4 packets.

It should be noted that with a regular traffic such as the one shown in figure 6.4, once the UPC has accepted a burst of maximum length sent at the PCR, it will discard cells from all the subsequent AAL5-PDUs sent at the PCR until the link has been idle for a

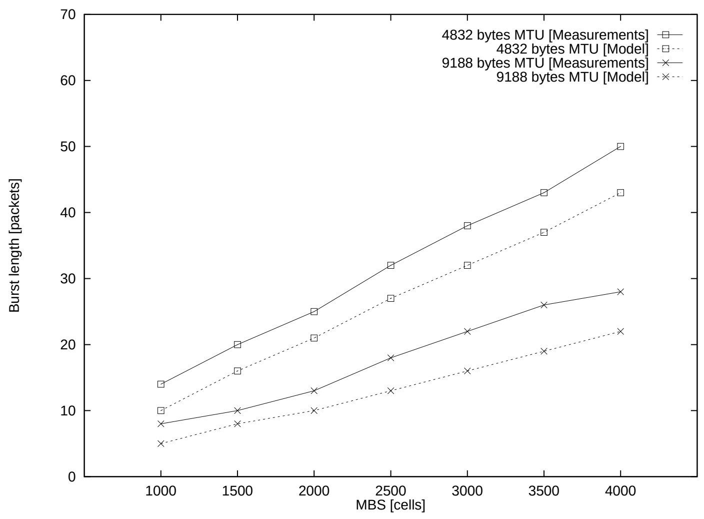

Figure 6.5: Comparison between the measurements and the packet train model

sufficient amount of time. The UPC will only accept a new entire AAL-PDU of c ATM cells sent at the PCR if the link has been idle for at least $c \times (T_{SCR} - T_{PCR})\mu$sec (e.g. 7.6 msec with the 9188 bytes MTU in our environment). The Inter-AAL-PDU time measured with our ATM adapters is much smaller than this value.

With the smaller MTU sizes, the agreement between the model and the measurements was not so good, especially with the 552 bytes MTU. This is probably because the workstations are more heavily loaded with these small MTU sizes. In this case, the assumption of a constant Inter-AAL-PDU time is not valid anymore.

Explanation for the large idle times

The large idle times, which resulted in a low throughput, are due to the particularities of the implementation of TCP's retransmission timer in SunOS and BSD-derived implementations [WS95]. In BSD-derived implementations, this retransmission timer is handled by the `tcp_slowtimo` routine which is called every 500 msec by the kernel [LMKQ89]. On each invocation, this routine checks for all the active TCP connections whether the retransmission timer has expired or not, and it invokes the `tcp_send` routine for each TCP connection whose retransmission timer has expired.

During our measurements, TCP transmitted a burst of packets. Cells from the last packets of this burst were discarded by the UPC, and TCP had to wait for the expiration of the retransmission timer to retransmit the lost packets. As the minimum value for TCP's retransmission timer is two 500 msec periods, the sender had to wait for approximately one second between two bursts of packets [BKD96a].

A look at the packet traces revealed that sometimes the idle time jumped to two seconds instead of one. These larger idle times were caused by the loss of one retransmitted packet. When one retransmitted packet is lost, TCP's retransmission timer is doubled (exponential back-off), and this explains the larger idle time. The loss of one retransmitted packet was caused by the corruption of the corresponding AAL5-PDU by unreassembled cells from previously transmitted AAL5-PDUs. These losses could have been avoided with the optional AAL5 reassembly timeout (RAS) defined in [ITU91], but this timeout was not supported by our ATM adapters [Cyp96].

We would like to point out that if we had used several TCP connections from the same workstation multiplexed into one VBR VC instead of a single TCP connection, the utilization of the VC would not have improved significantly. Assuming that these TCP connections start at random times, each connection will transmit one burst of packets and then the loss of the last packets of the burst will force it to wait for the expiration of its retransmission timer. Unfortunately, the retransmission timer will expire at the same time for several connections, and all these connections will start to retransmit together. A few packets will be retransmitted for each connection, before they will all be forced to wait for the expiration of their common retransmission timer. Obviously, such a synchronization among several connections will not lead to a better utilization of the VBR VC.

6.3.3 Possible Changes to the TCP Implementation

The TCP implementation used in SunOS 4.1.3_U1 is a production quality implementation which has been highly optimized during the years [MKK94]. It is representative of a wide range of currently available BSD-derived implementations. However, it does not contain all the TCP modifications which have been proposed recently. We discuss the impact of some of these modifications in the following sections.

Reducing the granularity of TCP's retransmission timer

The 500 msec granularity for the retransmission timer is obviously too large for the ATM environment. The current release of Solaris uses a 200 msec granularity for the retransmission timer [Ste94], and a 50 msec granularity for the delayed acknowledgment timer. We modified SunOS 4.1.3_U1 to use these smaller timer granularities. The measurements with these lower timer granularities (figure 6.6) showed that the throughput achieved by TCP almost doubled. With a MTU size of 552 bytes, TCP achieved a throughput of almost 6 Mbps with a MBS of 4000 cells, but only 2Mbps with a MBS of 1000 cells. With the larger MTU sizes, the TCP throughput was lower than 2 Mbps.

In SunOS 4.1.3_U1, any kernel timeout, including TCP's retransmission timer, depends on the `softclock` [LMKQ89] routine which is invoked by a clock interrupt every 10 msec. Thus, the minimum value for the retransmission timer in SunOS 4.1.3_U1 is 10 msec[2]. Figure 6.7 presents measurements performed with a 10 msec retransmission timer. For these measurements, we disabled the delayed acknowledgment timer in the receiver because this timer should ideally have a value smaller than the retransmission timer in order to avoid inaccuracies in the round-trip-time estimation.

From figure 6.7, one could conclude that the granularity of the retransmission timer should be set to 10 msec and that all the problems will be solved. This is not entirely true.

[2]We would like to point out that this value is much higher than the 100 μsec value used in some simulation studies [RF95]. To provide such a 100 μsec retransmission timer in BSD-derived implementations, the kernel should process one clock interrupt every 100 μsec. This would be very difficult with today's workstations and operating systems given the high cost of switching from user space to kernel space [Ous90]. Some kind of hardware support for the retransmission timer would probably be necessary to achieve such a 100 μsec retransmission timer with BSD-derived implementations.

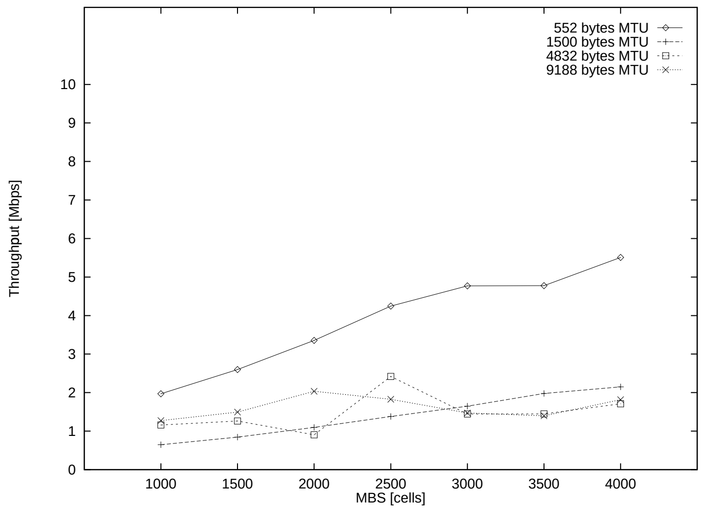

Figure 6.6: Measurements with a 200 msec granularity for the retransmission timer

The high throughput achieved with this low granularity is unfortunately accompanied by a huge number of retransmitted packets as shown by figure 6.8. This figure compares the number of retransmitted packets (reported by `netstat -s`) with the 500 msec, 200 msec and 10 msec retransmission timer granularities and a MTU size of 9188 bytes.

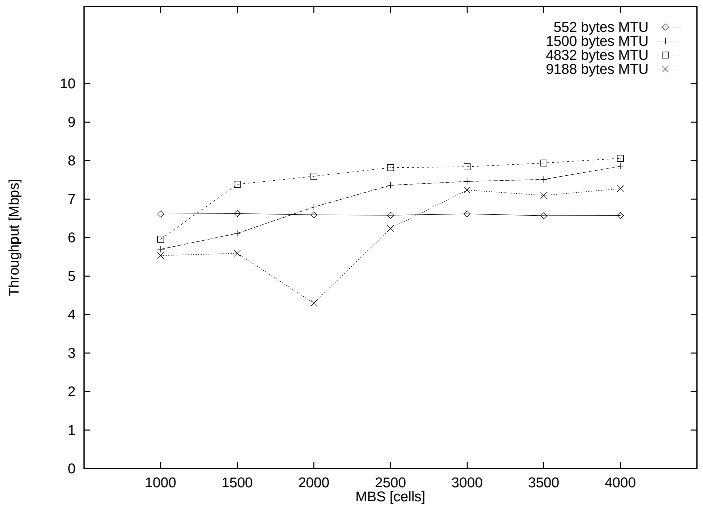

Figure 6.7: Measurements with a 10 msec granularity for the retransmission timer

We would like to point out that to transmit 4 MBytes of data, TCP must transmit 459 data packets with a MTU size of 9188 bytes. Thus, with the MBS set to 1000 cells and a 10 msec retransmission timer, 80% of the data packets are retransmitted. With the MBS set to 4000 cells, the percentage of retransmitted packets drops to 40% with the 10 msec retransmission timer, and 20% for the 200 and 500 msec retransmission timers. These high percentages of retransmissions show clearly that TCP, by itself, cannot adapt its behaviour to a VBR traffic contract in an ATM LAN.

The throughput drop which appeared with the MBS set to 2000 cells and a MTU size of 9188 bytes is linked with a kind of race condition. The TCP packet traces revealed an almost periodical behaviour. TCP first sent a burst of about 15 packets. Cells from the last 5 packets of this burst were discarded by the UPC. At that time, TCP's retransmission timer was set to 20 msec (the minimum value). The first retransmitted packet was corrupted with cells from previous incompletely reassembled packets, and thus discarded by the ATM adapter of the receiver. As this retransmission failed, the exponential back-off algorithm set the retransmission timer to 40 msec. The packet sent after the expiration of this timer was acknowledged by the destination. The sender performed slow-start, and successfully retransmitted the other 4 previously transmitted packets. The acknowledgments corresponding to these 4 packets did not update the value of TCP's retransmission timer as specified by Karn's algorithm [KP87]. Unfortunately the next packets were discarded by the UPC, and thus did not reach the destination. TCP's retransmission timer expired after 80 msec. The first retransmitted packet was discarded by the ATM adapter of the receiver as the corresponding AAL5-PDU was corrupted with cells from the previously transmitted, but unreassembled, AAL5-PDUs. After 160 msec of idle time, TCP attempted a new retransmission. This retransmission was successful, and TCP was able to send a burst of about 15 packets...

With the other MTU sizes and MBS, the retransmission timer did not reach such a high value. This throughput drop would have been avoided with the timestamp options specified in [JBB92] as when these options are used, TCP can update its retransmission timeout, even when it receives acknowledgments corresponding to retransmitted packets.

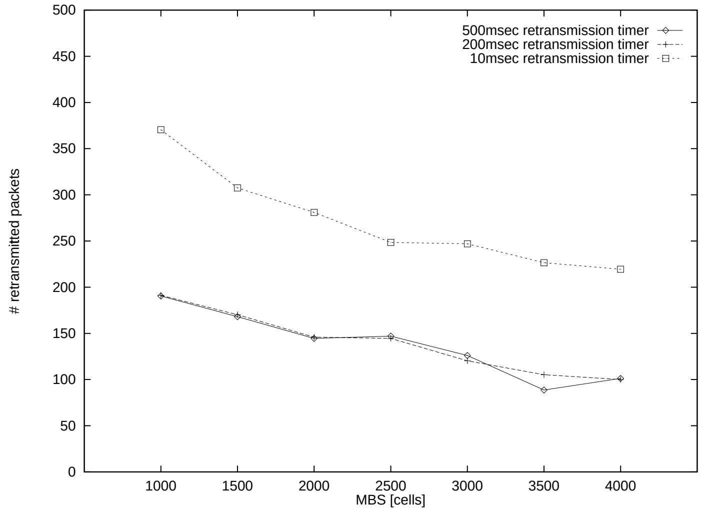

Figure 6.8: Number of retransmitted packets with the various retransmission timer granularities

Other retransmission strategies

As already mentioned, TCP's retransmission capabilities are probably one of its weakest points. In our environment, the fast retransmit algorithm was not sufficient to recover from the losses caused by the UPC, even when we set the duplicate acknowledgments threshold to 2. This is due to the fact that there was no long enough idle time between the last data packet transmitted and the retransmitted packet, and thus cells from the retransmitted packet were also discarded by the UPC.

The other enhancements to TCP's retransmission algorithm proposed recently like TCP Vegas [BOP94], [LK98] or the new selective acknowledgments option [MMFR96] would not have significantly changed the results of our measurements as all these strategies try to retransmit the lost packet as soon as possible. In a wide area network, however, these improved retransmission algorithms would probably be much more useful. In a LAN, a retransmitted packet will probably be discarded by the UPC since this packet will immediately follow a burst of packets. In a WAN, the larger round-trip-time implies that the link could be idle during a short period of time and thus a retransmitted packet could be declared conforming by the policing unit.

Traffic shaping

Our measurements have shown that TCP by itself cannot adapt its behaviour to a VBR traffic contract. To fully benefit from the VBR traffic contract with TCP, some additional support would be needed from the ATM driver or adapter. A solution could be to implement a dual-leaky bucket inside the ATM driver or adapter and to use this leaky bucket to shape the ATM traffic according to the VBR traffic contract. Our ATM adapters did not provide such leaky buckets in 1996, but when we set the cell rate of the ATM adapter to 10.000 Kbps[3](i.e. the SCR of the traffic contract), TCP achieved a good utilization of the VBR VC (figure 6.9) with the default 500 msec retransmission timer.

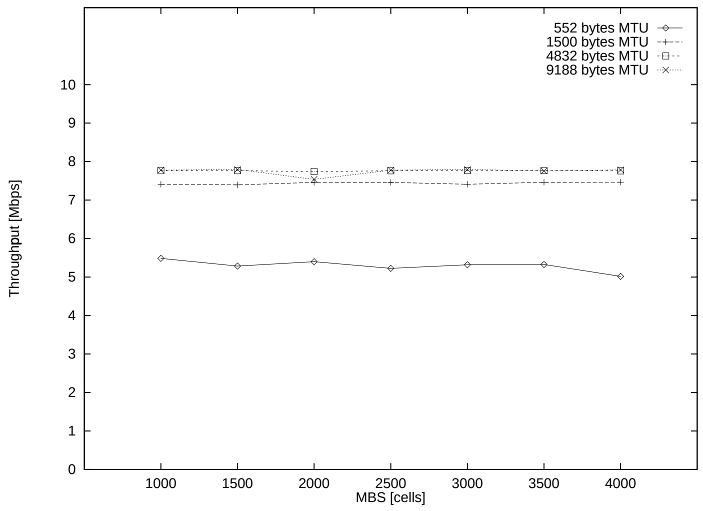

Figure 6.9: Measurements with the PCR of the ATM adapters set to 10.000 kbps

With the large MTU sizes, the throughput reached about 90% of the theoretical maximum if we take into account the overheads of the ATM cell header, AAL5 trailer and IP and TCP headers. There were no packet losses during the measurements with the 1500,

[3]Of course, by setting the cell rate of the ATM adapter to the SCR of the traffic contract, we do not benefit from burstiness allowed by the VBR traffic contract, but this was the only solution usable with our ATM adapters. In a production network with similar ATM adapters, it may be easier to use the CBR traffic contract instead of the VBR traffic contract when QoS guarantees are required.

4832 and 9188 bytes MTU sizes. This is in contrast with the measurements with the 10 msec retransmission timer (figure 6.7), where the percentage of retransmitted packets was very high. With a MTU size of 552 bytes, the achieved throughput is below 6 Mbps. This is due to the fact that with this MTU size, the receiving workstation has difficulties to sustain the load, and it drops some received packets. Indeed, 6 Mbps is the highest throughput we achieved with TCP, a MTU size of 552 bytes, a window size of 49152 bytes and the default 500 msec retransmission timer in our environment, even when the UPC was disabled.

This traffic shaping could also be implemented inside the ATM driver with a rate control at the AAL5-PDU level [4]. A regular flow of c cells long AAL5-PDUs (figure 6.4) will be conforming with a VBR traffic contract provided that the idle time (I_p) between successive AAL5-PDUs verifies [For93] the following equation :

$$c = 1 + \lfloor min(I_p - T_{SCR}, \tau_{SCR})/(T_{SCR} - T_{PCR}) \rfloor \tag{6.2}$$

It should be noted that the value of I_p given by equation 6.1 is valid when the source sends a regular flow of c cells long AAL5-PDUs at the PCR during an infinite amount of time. For example, with a MTU size of 9188 bytes (192 cells) and the VBR traffic contract used for our measurements, the minimum value of I_p is 7.6 msec. If the source sends its cells with a cell period larger than T_{PCR}, the minimum value of I_p will be smaller. If the source sends bursts of a few AAL5-PDUs, I_p can be smaller provided that the idle time between successive bursts is large enough. Indeed, the VBR traffic contract allows a source to send bursts of MBS cells at the PCR provided that the idle time between two successive bursts is at least equal to τ_{SCR} . If a rate control mechanism had to be implemented in SunOS, it could be controlled with a routine invoked by softclock every 10 msec [KB96]. If necessary, it could be possible to control both the cell rate of the ATM adapter and the Inter-AAL SDU time.

6.4 Suitability of VBR for internetwork traffic

In the previous section, we have studied the performance of TCP/IP over VBR when considering workstation traffic. We have shown that TCP had huge difficulties to benefit from the VBR service category. In this section, we use simulations to study the suitability of the VBR service category to efficiently support internetwork traffic.

For this, we consider a network environment where routers are interconnected through a wide area ATM network (figure 6.10). This ATM network is composed of E-3 (34 Mbps) links. The two ATM switches are linked with one E-3 link and ten routers are attached to this ATM backbone with 34 Mbps links. The delay on the link between the ATM switches is set to 10 milliseconds, while the delays on the links between the ATM switches and the routers is set to 2.5 milliseconds. The ATM switches use a PBS-based switch implementation and 16000 cells of buffers per port.

The ten routers are tail-drop routers with 700 KBytes of buffers and one ATM adapter. Each router is attached to one corporate LAN and we model each corporate LAN by

[4]Such a software-based shaper has also been proposed and implemented on top of an AAL5-driver in [LAH97] and in a native ATM transport protocol in [AKS96].

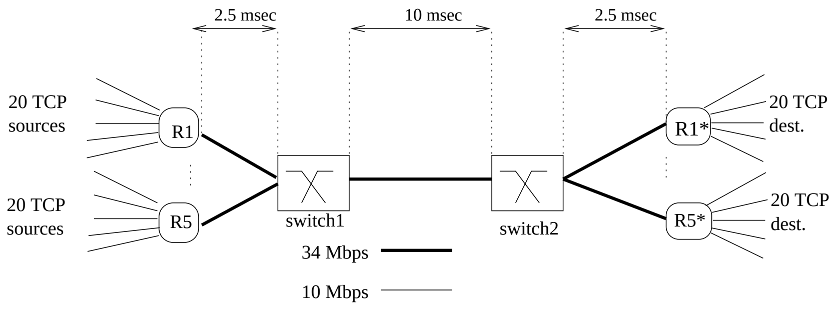

Figure 6.10: Simulation environment for internetwork traffic

considering that twenty workstations are attached to the router with point-to-point 10 Mbps links[5]. We only consider one-way traffic and assume that all the TCP sources are attached to the routers on the left side of figure 6.10 (R1-R5) and that the TCP destinations are attached to the routers on the right side (R1*-R5*) of the figure. The routers are grouped by pairs and all the workstations attached to one router send traffic to workstations attached to its companion router. For example each of the workstations attached to R1 send traffic to a workstation attached to R1*. The TCP sources are file servers which continuously send a 1 MByte file. The TCP window sizes of the workstations is set to 64 KBytes and the MSS is set to 1460 bytes (the default MSS size for TCP over Ethernet). We use a 200 msec granularity for the retransmission timer and a 50 msec granularity for the delayed acknowledgment timer.

One VBR VC is established between two companion routers. All the VBR VCs have the same PCR (34 Mbps), but we vary the SCR as shown in table 6.1. The VBR traffic contracts of these VCs are enforced by a UPC which tags the non-conforming cells at the ingress of the network.

Table 6.1: Internetwork traffic : characteristics of the VBR VCs

VC	PCR	SCR	Router-router delay
R1-R1*	34 Mbps	2 Mbps	15 msec
R2-R2*	34 Mbps	4 Mbps	15 msec
R3-R3*	34 Mbps	6 Mbps	15 msec
R4-R4*	34 Mbps	8 Mbps	15 msec
R5-R5*	34 Mbps	10 Mbps	15 msec

We performed two types of simulations in this environment. We first consider a worst-case scenario in section 6.4.1 by setting the $PBS_{threshold}$ to zero. Then, we consider a less congested network in section 6.4.2 and allow the $PBS_{threshold}$ to be positive.

[5]These point-to-point links are used to model a 10 Mbps switched-Ethernet LAN.

6.4.1 Simulations with congested switch buffers

When the ATM switches cannot accept any CLP=1 cells, the simulations show that the workstations attached to each router have several difficulties to benefit from the bandwidth reserved on the VBR VCs in the ATM network. Figure 6.11 compares the total goodput of the twenty workstations attached to each router with different values for the MBS with the "ideal total goodput". The "ideal total goodput" is the SCR of the VBR VCs minus the ATM, AAL, IP and TCP overheads. The "ideal goodput" thus assumes that no packets are retransmitted.

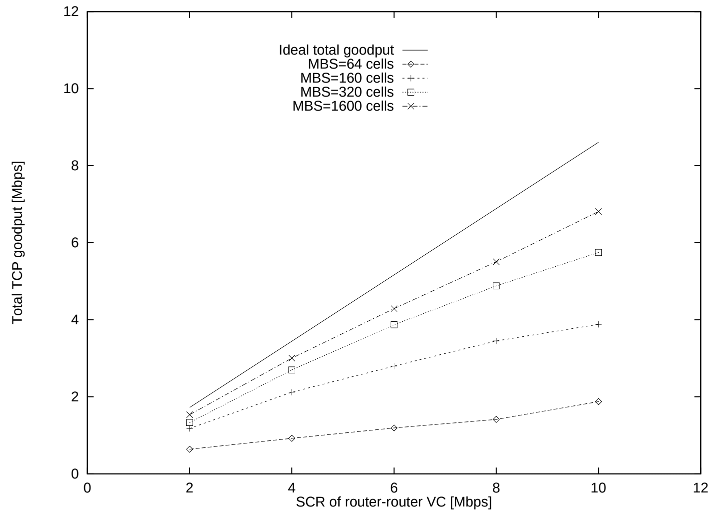

Figure 6.11: Internetwork traffic with congested switch buffers

With a MBS corresponding to one maximum size IP packet (32 cells in our environment), the data transfer was almost impossible for all the workstations. The goodput achieved by the workstations attached to the different routers in this case was very low and all the workstations transmitted a few data packets and were waiting for the expiration of their retransmission timeout for the remainder of the time. With an MBS corresponding to two IP packets (64 cells), the performance was much better, but still far from satisfactory. The workstations attached to the 2 Mbps router[6] achieved a total goodput of 450 Kbps while the total goodput for the workstations attached to the 10 Mbps router reached 2 Mbps. With a much larger MBS (e.g. 1600 cells), the workstations attached to the 2 Mbps router were almost able to efficiently utilize their VBR VC, but the total goodput for the workstations attached to the 10 Mbps router was still lower than the capacity of their VC.

When comparing these simulations with the measurements discussed in section 6.3.1 it appears that the VBR service category provides a better, but still not satisfactory, performance with internetwork traffic than with workstation traffic. The measurements have shown that the main reason for the low performance of TCP in the ATM LAN we considered was the TCP retransmission timer. When the traffic on the ATM VC comes from several independent workstations, the inefficiency of the TCP retransmission timer is

[6]By 2 Mbps router, we mean the router (R1) with a 2 Mbps ATM VC.

less severe since one workstation can be active while another is waiting for the expiration of its retransmission timer.

6.4.2 Simulations with lightly congested switch buffers

When the value of the $PBS_{threshold}$ is sufficiently large to accept some CLP=1 cells, the performance of TCP/IP is much better. Figure 6.12 shows the total goodput achieved by the workstations attached to each router with a $PBS_{threshold}$ set to 6000 cells. The total TCP goodput is much better than when $PBS_{threshold}$ was set to zero, but the simulations reveal an interesting behaviour. The workstations attached to the 2 Mbps router always achieve a much higher goodput than their ideal total goodput, while the workstations attached to the 10 Mbps router cannot reach their ideal total goodput. This behaviour is obviously not desirable.

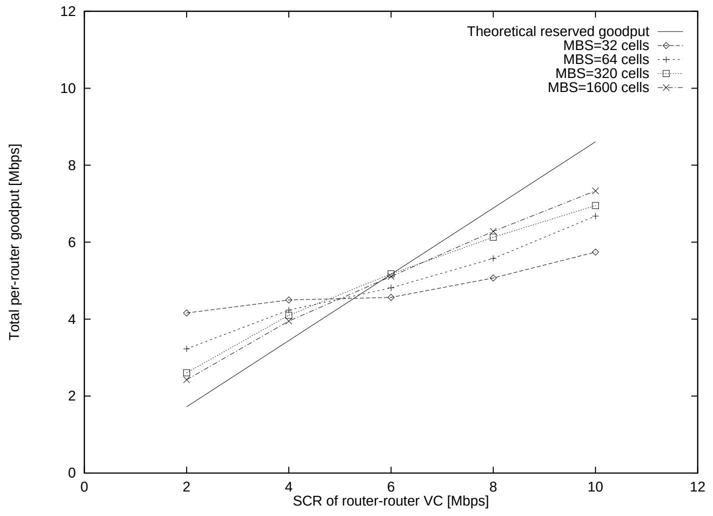

Figure 6.12: 5 routers scenario with VBR.3

The relatively low performance of the VBR.3 conformance definition and the PBS switch implementation when carrying internetwork traffic is mainly due to two factors. The first factor is that, as already shown by section 6.4.1, there is some mismatch between internetwork traffic and a VBR traffic contract enforced by a VBR UPC. This mismatch is due to the high burstiness of the internetwork traffic. Due to this mismatch, a large number of ATM cells can be tagged by the VBR UPC, even if the long term average rate of the VC is below the SCR of the VC. For example if we consider the simulation shown in figure 6.12 with an MBS of 32 cells, then 30 % of the ATM cells sent on the 10 Mbps VC were tagged by the VBR UPC, although the long term average rate of the cells coming out of this router was close to 7.8 Mbps. Since the VBR UPC tags a large fraction of the cells sent on the router-router VCs, a lot of CLP=1 cells travel the network. Although 30 Mbps out of 34 Mbps were "reserved" for the CLP=0 cells on the link between the two switches, the simulations showed that this link carried much more than 4 Mbps of CLP=1 cells. For example with a MBS of 32 cells, 50 % of the bandwidth on the inter-switch link was used by the CLP=1 cells.

The second factor is that neither the VBR UPC nor the PBS switch implementation are aware of the AAL5-PDU boundaries. This means that the VBR UPC may tag several cells inside an AAL5-PDU without tagging all the cells of this PDU. Since the PBS switch implementation discards only CLP=1 cells when it is congested, the combination of the VBR UPC and the PBS switch implementation implies that AAL5-PDUs will be partially discarded when congestion occurs. However, the (non-discarded) CLP=0 cells from these AAL5-PDUs will still be carried on the ATM links and thus will waste bandwidth during periods of congestion. This problem was already responsible for a reduction of the performance in the simple network environment we studied. For example, during the simulations with an MBS set to 64 cells, the workstations attached to the 8 Mbps router on the right side of figure 6.10 received on average 1.7 corrupted AAL5-PDUs and on average 28 correct AAL5-PDUs every second. In larger networks, carrying incomplete AAL5-PDUs would become a major source of inefficiency since the (CLP=0) cells from these AAL5-PDUs will be carried uselessly on a potentially large number of links.

A possible solution to reduce the inefficiency due to the transmission of cells from incomplete AAL5-PDUs would be to add some packet discard mechanism to the PBS switch implementation (e.g. PPD). However, such a packet discard mechanism is, at least partially, in contradiction with the low CLR associated with the CLP=0 cells since CLP=0 cells would be discarded by such a mechanism. The ATM Forum Traffic Management specification [For96c] defines a generic frame discard mechanism, but does not interpret how such a mechanism should or could be used with the VBR service category and what are its possible implications on the QoS commitments for the CLP=0 cells. The ITU-T recommendations [ITU96c] do not currently define a frame discard mechanism. However, such a mechanism has been under study since May 1996 and the current I.371 living list document [SG198] notes concerning the utilization of a frame discard mechanism with the VBR service category that "*the combined used of tagging, selective cell discard and frame discard on the same connection may not be efficient*".

6.5 Related work

Several researchers have studied the performance of TCP with the VBR service category. Several papers discuss the impact of the VBR UPC on the performance of TCP by simulations [EARH95, SST95, Mis96, Mee97, SD97]. The measurements described in the first part of this chapter [BKD96b] were performed slightly after the simulations described in [EARH95, SST95, Mis96]. Other papers have studied the performance of TCP with the VBR.2/3 conformance definitions [PR95, HV98a] and two papers analyse measurements [AHK$^+$97, AHK$^+$98].

In [SST95, Mis96], the impact of a VBR UPC on the TCP goodput is analyzed with an analytical model and by simulations. This paper only considers workstation traffic. The simulations show, like our measurements, that TCP has several difficulties to efficiently utilize a VBR VC subject to strict policing. [Mis96] shows that the low performance of TCP is mainly due to the large granularity of the TCP retransmission timer. This is in line with our measurements. Then, [Mis96] discusses two possible modifications to the VBR UPC. The first one is similar to the F-GCRA which is used for the GFR service category. [Mis96] shows that this UPC does not provide a significant improvement over the classical GCRA. The second modified GCRA proposed in [Mis96] tries to avoid discarding two successive AAL5-PDUs by discarding some AAL5-PDUs earlier than a classical GCRA. This modified GCRA provides an improvement over the classical GCRA, but the simulations presented in [Mis96] show that the performance of this modified GCRA is sensitive to the choice of its parameters. [Mis96] does not consider internetwork traffic.

[EARH95] also analyses the impact of a VBR UPC on workstation traffic. The simulations discussed in this paper show a negative impact of the VBR UPC on the performance of TCP. In order to improve the performance of TCP, [EARH95] proposes to replace the slow-start and congestion avoidance mechanisms by a packet-based leaky bucket when TCP is used with a VBR ATM VC. Simulations show that this solution provides a much better performance than the default slow-start and congestion avoidance mechanisms. However, the solution proposed in [EARH95] is rather "ad hoc" and not really usable in practice since it assumes that TCP is aware of the traffic contract of the underlying ATM VC and that each TCP connection is carried in a single ATM VC. These two conditions (especially the first one) are not always met in reality. A software implementation of a packet-based shaping algorithm inspired by [EARH95] is discussed in [LAH97]. [AKS96] also proposes to include a packet-based shaping algorithm in a transport protocol designed to be used in native ATM networks.

[Mee97] also studies the interactions between TCP and a packet-based policing unit (e.g. F-GCRA). The simulations discussed in this paper also show that when a single TCP connection is carried in each ATM VC, the TCP performance is severely degraded when the policing unit discards the non-conforming packets.

[SD97] analyses the effect of several modified GCRAs on Internet traffic by simulating the effect of these algorithms on ATM-level traces of an Internet backbone. The simulations discussed in this paper show that adding some AAL5 awareness to the GCRA reduces the percentage of AAL5-PDUs which are declared (at least partially) non-conforming by the policing unit.

[PR95] studies the performance of workstation traffic with the VBR.2/3 service category. To improve the performance of TCP in these conditions, [PR95] proposes to modify TCP to intelligently use the CLP bit. [PR95] proposes to use AAL5-PDUs composed of CLP=1 cells to "probe" the state of the network when TCP increases its window size and show that this modification provides some benefit in a WAN environment.

In [HV98a], simulations are briefly used to compare the performance of the VBR service category (with the VBR.3 conformance definition) with the GFR service category (with a simple implementation supporting the GFR.2 conformance definition) to carry internetwork traffic. Concerning VBR.3, [HV98a] shows that when the amount of reserved bandwidth ($\sum_i SCR[i]$) inside the ATM backbone is low, then VBR.3 provides a satisfactory performance with internetwork traffic, but this performance decreases when the amount of reserved bandwidth approaches the capacity of the links of the ATM backbone.

[AHK$^+$97] and [AHK$^+$98] study the impact of a VBR UPC on the performance of TCP. By using several workstations and a delay emulator, they are able to perform measurements with both workstation and internetwork traffic in MAN and WAN environments. They first try to experimentally determine an optimum VBR traffic contract (MBS and SCR) for TCP traffic. Their measurements show that the optimum VBR traffic depends on the MSS, MBS, round-trip-time and number of TCP connections on each VC. As a rule of thumb, they propose to use $MBS = \lfloor (WIN \times N)/MSS \rfloor$ and $SCR = MBS/RTT$ where WIN is the window size of the TCP connections (supposed equal for all the TCP connections), N the number of TCP connections carried in the ATM VC, MSS the MSS of all the TCP connections and RTT the round-trip-time of the N TCP connections. Based on this optimum VBR traffic contract, they then study the impact of non-optimum MBS and SCR values on the TCP performance. Their measurements show that the TCP performance decreases quickly when the MBS is reduced when each ATM VC carries one TCP connection but that the performance reduction is less severe when four TCP connections are carried by one ATM VC.

The measurements discussed in [AHK$^+$97] and [AHK$^+$98] are complementary to the measurements we discussed in section 6.3.1 since we considered a LAN environment while [AHK$^+$97] only considers MAN and WAN environments. Furthermore, we have also studied several modifications to the TCP implementations. This aspect is not addressed in [AHK$^+$97, AHK$^+$98].

6.6 Conclusion

In this chapter, we have identified several important characteristics of the VBR service category with respect to TCP/IP traffic. We have first shown that, when considering workstation traffic, TCP had huge difficulties to adapt its behaviour with the VBR.3 conformance definition when all the CLP=1 cells were discarded inside the network due to congestion. We have also shown that although it was possible to improve the performance of TCP by reducing the granularity of the retransmission timer, this "solution" had an unacceptable hidden cost in terms of retransmitted packets. Due to this problem, the utilization of the VBR service category in ATM LAN to provide services with a minimum guaranteed bandwidth is questionable [BKD96b].

Then we have studied the suitability of the VBR service category to efficiently support internetwork traffic in ATM WANs. We have shown that, in this case, the performance is highly dependent on the MBS of the ATM VCs. When the MBS is small (corresponding to a few maximum-sized IP packets), the workstations attached to the routers have difficulties to efficiently utilize the minimum guaranteed bandwidth of their router's ATM VC. With a larger MBS (a few tens of maximum-sized IP packets), the performance is much better, but still far from perfect.

We have identified two main reasons for the low performance of the VBR service category. The main reason is that internetwork traffic is more bursty than the traffic contract enforced by a UPC at the ingress of the network. Due to this burstiness, the UPC may tag many ATM cells although the long term average rate on the ATM VC is smaller than the SCR.

The second reason is that the GCRA used by the UPC at the ingress of the network and the PBS switch implementation are not aware of the AAL5-PDU boundaries and thus discard individual ATM cells instead of complete AAL5-PDUs. Although it would be possible to utilize packet discard mechanisms with the VBR GCRA and PBS, we have shown that the standards were not clear about whether packet discard mechanisms were acceptable for the VBR service category or not.

Chapter 7

TCP and the GFR service category

7.1 Introduction

The previous chapters have identified the respective drawbacks of the CBR, ABR and VBR service categories to efficiently support TCP/IP services with a minimum guaranteed bandwidth. In this chapter, we present a detailed simulation study of the performance of TCP/IP with the GFR service category.

This chapter is structured as follows. We first present in section 7.2 the two switch implementations which have been proposed by the proponents of the GFR service category. These two switch implementations can be considered as a baseline to which other implementations should be compared. After this presentation, we study the performance of these two implementations, first with workstation traffic in section 7.3 and then with internetwork traffic in section 7.4. Based on the simulations discussed in these two sections, we identify several drawbacks of these two switch implementations in section 7.5.

7.2 Proposed GFR switch implementations

Two switch implementations were proposed in the first description of the GFR service category [GH96]. These two implementations were later retained as example implementation for the baseline ATM Forum text [Ken98]. The first implementation, which we will name Double-EPD in this thesis, can be used in simple FIFO-based switches. It can be considered as a backward compatible solution for legacy switches. The second implementation, which we will name per-VC threshold and scheduling in this thesis, is much more complex.

7.2.1 Double-EPD

The FIFO-based switch implementation proposed in [GH96] is an adaptation of the Partial Buffer Sharing [KHBG91] buffer acceptance algorithm which is frequently used to support VBR.2 and VBR.3 VCs in ATM switches. It only supports the GFR.2 conformance definition. This FIFO-based switch implementation is an AAL5-aware buffer acceptance algorithm which relies on two buffer thresholds. These two thresholds are the Low Buffer Occupancy (LBO) and the High Buffer Occupancy (HBO) thresholds. The highest threshold (HBO) is identical to a classical EPD threshold [RF95]. The lowest threshold (LBO) is used to limit the amount of non-eligible (CLP=1) AAL5-PDUs inside the buffer. The LBO threshold is used as an EPD threshold for the CLP=1 AAL5-PDUs. When the queue occupancy of the buffer is above the LBO threshold, then the newly arriving CLP=1 AAL5-PDUs are not accepted anymore in the buffer (but the newly arriving CLP=0 AAL5-PDUs are still accepted provided that the queue occupancy is below the HBO threshold).

The Double-EPD implementation is shown graphically in figure 7.1. All the GFR VCs are multiplexed in a single FIFO buffer which is attached to an output link with a bandwidth equal to PCR.

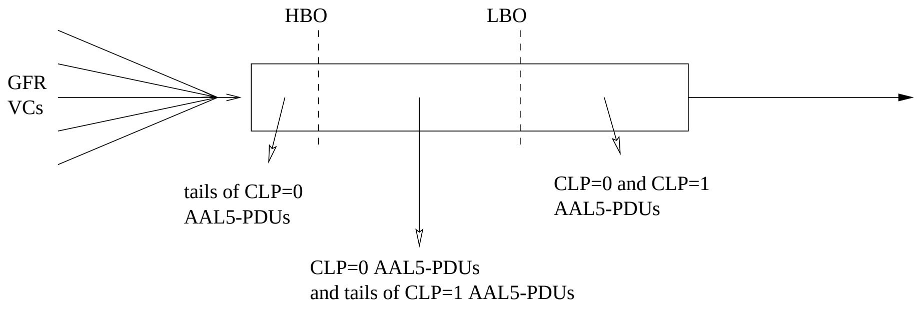
HBO
LBO
GFR VCs
tails of CLP=0 AAL5-PDUs
CLP=0 and CLP=1 AAL5-PDUs
CLP=0 AAL5-PDUs and tails of CLP=1 AAL5-PDUs

Figure 7.1: The Double-EPD switch implementation

7.2.2 Per-VC threshold and scheduling

This implementation combines a buffer acceptance algorithm with a per-VC scheduler. It was first proposed in [GH96]. It provides the bandwidth guarantees required to support the GFR service category by maintaining one logical queue for each GFR VC and by serving these queues with a WFQ-like scheduler at a rate at least equal to their MCR. The utilization of this scheduler guarantees that, when active, each VC will be allocated its reserved bandwidth as well as some fairshare of the available excess bandwidth. Many schedulers have been proposed in the literature [Zha95]. For this work, we have chosen to use Virtual Spacing [Rob94], which is equivalent to SCFQ [Gol94] with fixed-size packets. The main advantage of Virtual Spacing over other WFQ-like schedulers is that it appears to be implementable at broadband speeds [RGB96, BSZ97].

The Virtual Spacing algorithm maintains one state variable for each VC (VS_i) and a global state variable (*Spacing Time*) per output buffer. A weight (r_i) is associated with each VC. For the GFR service category, this weight will be equal to the MCR of the VC. The Virtual Spacing algorithm associates a timestamp to each arriving cell as follows :

- On a cell arrival from VC i
    1. $VS_i \leftarrow \max\{\text{Spacing time}, VS_i\} + 1/r_i$
    2. timestamp the cell with the value of VS_i
- All the cells are served in increasing order of their timestamp

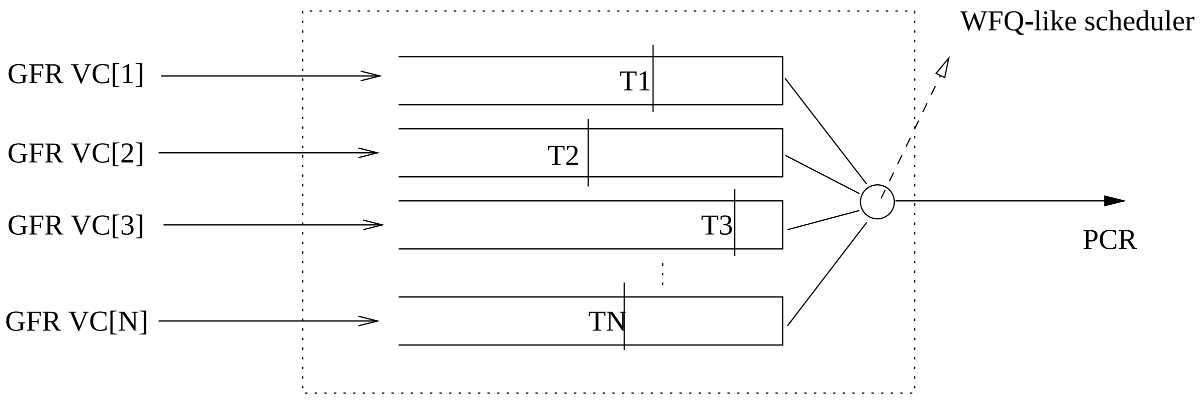

Figure 7.2: The per-VC threshold and scheduling switch implementation

- *Spacing time* is set to the timestamp of the last served cell

In addition to the per-VC scheduler, this implementation also relies on a buffer acceptance algorithm (figure 7.2). This algorithm requires a global counter for the buffer occupancy and one counter for the occupancy of each per-VC queue. It then uses two buffer thresholds (HBO and LBO) and one threshold for each per- VC queue (T_i). [GH96] proposes to set these per-VC thresholds to the MBS of the GFR traffic contract for each VC. The LBO threshold has the same role as with the FIFO-based implementation : the arriving CLP=1 AAL5-PDUs are only accepted if the buffer occupancy (i.e. the total number of cells in all the per-VC queues) is below the LBO threshold. The HBO threshold is used together with the per-VC thresholds. When the buffer occupancy is between the LBO and the HBO thresholds, then all the arriving CLP=0 AAL5-PDUs are accepted. When the buffer occupancy is above the HBO threshold, then an arriving CLP=0 AAL5-PDU is accepted only if the queue occupancy of its VC is below its T_i threshold.

7.3 Performance with workstation traffic

In this section, we present a detailed simulation study of the performance of the Double-EPD and per-VC threshold and scheduling implementations with workstation traffic.

7.3.1 The simulation model

Our simulations were performed with the STCP simulator which is described in appendix B. We consider only one-way traffic and greedy TCP sources in this section. Each TCP source is driven by an application which opens a TCP connection, sends 10 MBytes of data, closes the TCP connection, waits for some idle time, opens a new TCP connection, performs a new 10 MBytes transfer... These idle times are exponentially distributed with a mean duration of 0.1 second. They are used to introduce some randomness in the simulations to avoid synchronization effects. Unless otherwise noted, the simulation results reported in this paper correspond to average values for a simulation corresponding to 250 seconds of simulated time. This is much longer than most of the simulations reported in the literature and gives us a high confidence in our simulation results.

The ATM switch is modeled as a non-blocking output buffered switch. In this switch, the only cause of congestion is the possible overflow of its output buffers. For our simulations, we used a small ATM network (figure 7.3). All the links in this network have a bandwidth of 155 Mbps (365566 cells/sec). For the LAN simulations, we use a delay of 10 μsec on the UNI links and a delay of 100 μsec on the NNI link. For the WAN simulations, we use a delay of 2.5 msec on the UNI links, and a delay of 10 msec on the NNI link. We consider that one of the sources (the privileged source) is more important than the other sources (called the background sources). The privileged and the background sources differ only in their respective GFR traffic contracts. All the background sources use a GFR traffic contract with a low MCR. The privileged source uses a larger MCR (up to 50% the NNI link bandwidth). The MFS is set to 200 cells for both types of sources, and the PCR is set to the line rate (365566 cells per second) for both types of sources. In all the simulations with the FIFO-based switches, the MBS was set to allow a burst of two MFS-sized AAL5-PDUs sent at PCR to be found eligible for the minimum guaranteed bandwidth. This is in line with the initial proposal for the GFR service category [GH96].

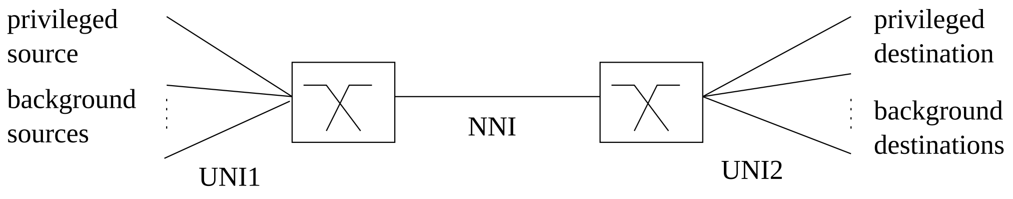

Figure 7.3: The simulated ATM network

7.3.2 LAN simulations with Double-EPD

Instead of implementing the F-GCRA shown in figure 3.7, we choose to perform the tagging inside the model of the ATM adapters. This choice was motivated by the fact that the F-GCRA was not completely defined by the ATM Forum when we performed the simulations with the Double-EPD implementation [Bon97], but also because the endsystem is a better place to tag the cells on a per AAL5-PDU basis than inside a UPC. The AAL5-PDU tagging was performed by the algorithm shown in figure 7.4 (where $T_{MCR} = 1/MCR$, $T_{PCR} = 1/PCR$, $\tau_{MCR} = (2 \times MFS - 1) \times (T_{MCR} - T_{PCR})$). We configured the tagging in the ATM adapters to allow a burst of $2 \times MFS$ cells CLP=0 cells to be sent at PCR by the adapters. With MFS-sized AAL5-PDUs, the tagging performed by the ATM adapters is equivalent to F-GCRA[T_{MCR}, $MFS \times (T_{MCR} - T_{PCR})$], but the tagging performed by the ATM adapters may accept some additional small CLP=0 AAL5-PDUs with variable-sized AAL5-PDUs. Both algorithms should tag the same percentage of AAL5-PDUs although they may not tag exactly the same AAL5-PDUs.

The TCP sources and destinations were configured with send and receive socket buffers (i.e. maximum window sizes) of 192 kBytes. The delayed acknowledgements were disabled

```
Beginning of AAL5-PDU transmission at time
$t_a$ :

if ($t_a$ +$(AAL5\_PDU_{length} - 1) \times T_{PCR} \geq$
    $max(t_a, TAT_{MCR}) + (AAL5\_PDU_{length} - 1) \times T_{MCR} - \tau_{MCR}$ )
    {
        $TAT_{MCR} = max(t_a, TAT_{MCR}) + AAL5\_PDU_{length} \times T_{MCR}$
        /* send whole AAL5-PDU at PCR with CLP=0 cells */
    }
    else
    {
        /* send whole AAL5-PDU at PCR with CLP=1 cells */
    }
```

Figure 7.4: AAL5-PDU tagging in the ATM adapters

on each TCP destination. The granularity of the retransmission timer was set to 0.2 seconds. These two timer granularities are lower than the default values used by 4.4 BSD Lite [WS95], but they correspond to the values used by commercial TCP implementations (e.g. Solaris 2.x). The fast retransmit mechanism was enabled and the retransmission threshold was set to the suggested default (3 duplicate acks). The TCP maximum segment size was set to 9140 bytes. The main TCP parameters used for this first simulation are summarised in table 7.1. Throughout the remainder of this thesis, we will refer to this set of parameters as $TCP_{default}$.

Table 7.1: TCP parameters for $TCP_{default}$

Parameter	Value
retransmission timer	0.2 seconds
fast retransmit threshold	3 duplicate acks

Before discussing the GFR simulations, it is interesting to first examine an artificial UBR simulation that could be considered as a baseline for the GFR simulations. For this artificial UBR simulation, we considered an ATM LAN similar to the one shown in figure 7.3. In this LAN, the UNI1 and NNI links used a PCR of 365566 cells per second, while the UNI2 links had a lower PCR. We used 9 background sources, and the PCR of the UNI2 links connected to the background destinations was set to 20000 cells per second, while the PCR of the UNI2 link connected to the privileged destination varied from 20000 to 180000 cells per second (table 7.2).

The PCR of all the sources was set to the UNI1 PCR (365566 cells per second).The ATM switches had a 8192 cells buffer per output port, and the EPD threshold was set to 7168 cells. This simulation scenario is completely artificial since the sum of the bandwidth on the UNI2 links is always smaller than the bandwidth on the NNI link and thus the NNI

Table 7.2: Link characteristics for the UBR reference scenario

Link	PCR
sources-switch1	365566 cells per second
switch1-switch2	365566 cells per second
switch2-privileged destination	20000 to 180000 cells per second
switch2-background destinations	20000 cells per second

link is not congested. We use this artificial scenario to verify whether the TCP sources are able to "discover" and utilize efficiently the bandwidth available on the UNI2 links. The simulations performed with this artificial scenario showed that $TCP_{default}$ was able to completely utilize the bandwidth available on the UNI2 links, both for the privileged and the background sources (figure 7.5).

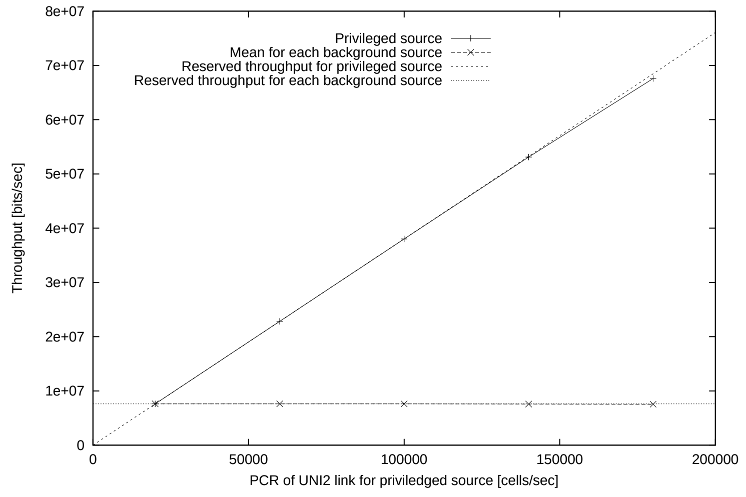

Figure 7.5: $TCP_{default}$ throughput with bottleneck on UNI2 links

Figure 7.5 shows the mean user-level throughput (i.e. TCP goodput) achieved by the privileged source as well as the mean throughput achieved by each background source with $TCP_{default}$. In addition to these simulation results, figure 7.5 also recalls the amount of reserved bandwidth for both the privileged and each background source. The reserved bandwidth shown in figure 7.5 accounts for the protocol overhead (i.e. ATM, TCP and IP headers, TCP timestamp option and AAL5 trailer), and thus can be considered as the "application-level" reserved throughput.

For our first GFR simulations, we used 9 background sources and one privileged source. Each background source had a reserved bandwidth (MCR) of 20000 cells per second. Thus, 50% of the NNI link bandwidth was reserved for the background sources. The MCR of the privileged source varied from 20000 to 180000 cells per second. The MFS of all the sources was set to 200 cells (i.e. slightly more than the default CPCS-PDU size for IP over ATM). The PCR of all the sources was set to the line rate (365566 cells per second). The Double-EPD switches were configured with a buffer capacity of 8192 cells per output port.

This output queue size corresponds to the buffer sizes used by current commercial ATM switches. The HBO and LBO thresholds were set to 7168 and 6144 cells respectively.

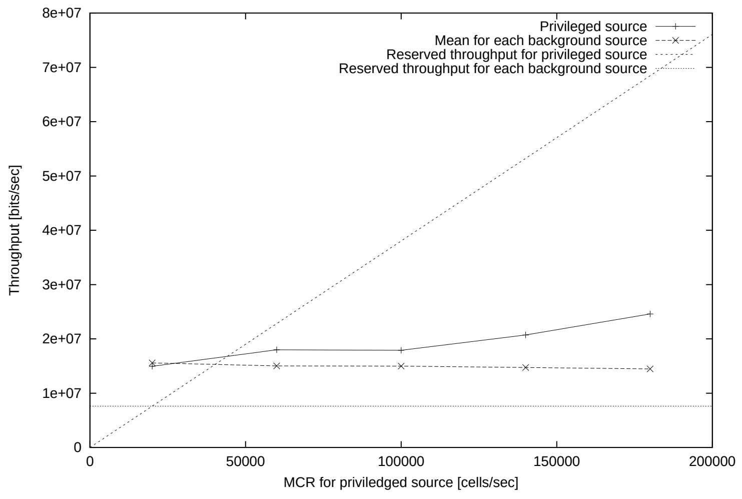

Figure 7.6: $TCP_{default}$ throughput for privileged and background sources

Figure 7.6 clearly shows that the privileged source has a lot of difficulties to actually use its reserved bandwidth. With its MCR set to 100000 cells per second the privileged source achieved only about 18 Mbps, less than 50% of its reserved bandwidth. With its MCR set to 180000 cells per second, the privileged source only achieved a throughput of about 25 Mbps, only one third of its reserved bandwidth.

A look at the simulation traces revealed two reasons for the low performance of $TCP_{default}$. The first one is the large granularity of the TCP retransmission timer with $TCP_{default}$. In a low speed network, a delay of a few hundred milliseconds is not important, but in a high speed network, it may correspond to the transmission of a few megabytes of data. During the simulations reported in figure 7.6, and with the MCR of the privileged source set to 180000 cells per second, the privileged source had to retransmit about 200 KBytes during each 10 MBytes transfer, and the retransmission timer expired on average slightly less than four times during each transfer. As the minimum value of the retransmission timer is equal to the retransmission timer granularity, the average idle time may easily reach one second per 10 MBytes transfer, almost as long as the time to transfer 10 MBytes without losses at a rate of 180000 cells per second. Another factor which limits the throughput of the privileged source is the low average value of the congestion window (figure 7.7) combined with the high average occupancy of the output buffer of the bottleneck switch.

These expirations of the retransmission timer were of course caused by packet losses in the bottleneck switch. With the Double-EPD switch implementation, CLP=1 AAL5-PDUs are discarded when the occupancy of the output buffer of the bottleneck switch is larger than the LBO threshold, and CLP=0+1 AAL5-PDUs are discarded when the size of the output buffer of the bottleneck switch is larger than the HBO threshold. A look at the simulation traces revealed that the output buffer of the bottleneck switch did not reach the HBO threshold, and that only CLP=1 AAL5-PDUs were discarded at the

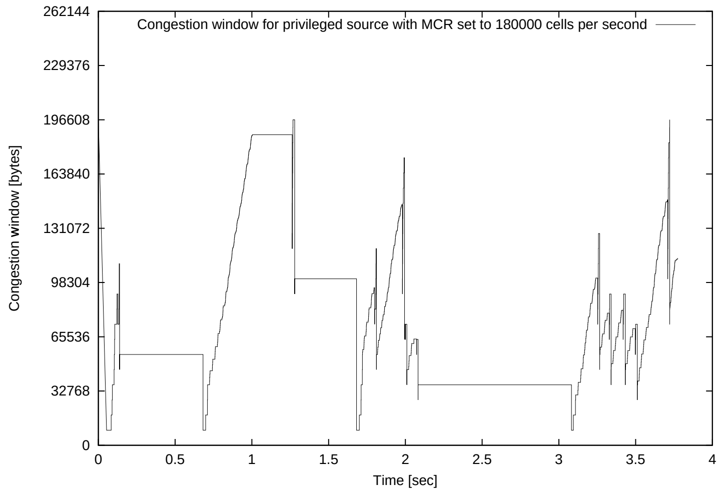

Figure 7.7: $TCP_{default}$: sample congestion window trace

bottleneck switch. With its MCR set to 100000 cells per second, slightly less than 50% of the AAL5-PDUs sent by the privileged source were tagged, while with its MCR set to 180000 cells per seconds 30% of its AAL5-PDUs were tagged. This large percentage of tagged AAL5-PDUs explains why the privileged source suffered from packet losses even though it was unable to efficiently use its reserved bandwidth.

In the following sections, we will first study whether it is possible to improve the performance of TCP by changing some of its parameters (i.e. timer granularities and retransmission mechanisms). Then, we will change the background load, the number of background sources and the LBO threshold to see if they have an influence on the simulation results. Finally, we will look at the influence of TCP's maximum segment size.

TCP timers and retransmissions

Several TCP mechanisms and parameters may influence the TCP throughput in our environment. These mechanisms are mainly the timer-based retransmissions and the fast retransmit algorithm.

The timer-based retransmission is essential to TCP, but its negative impact on the achieved throughput may be reduced by setting its granularity to a low value. In most BSD-derived implementations, the minimum value of the retransmission timer cannot be lower than the period of the real-time hardware clock. In Unix variants, the period of this clock is typically set to 10 milliseconds. As the ATM layer guarantees the in-sequence delivery of the data, another possibility to lower the impact of the retransmission timer is to reduce the value of the fast retransmit threshold down to one duplicate acks. Throughout the remainder of this thesis, we will use the acronym TCP_{fast} for a TCP source which uses the parameters shown in table 7.3.

Figure 7.8 shows the throughput achieved by the privileged and the background sources with TCP_{fast} and a 192 KBytes window. Surprisingly, the throughput achieved with TCP_{fast} by the privileged and the background sources in these conditions is slightly lower than the throughput achieved by $TCP_{default}$. With TCP_{fast}, the privileged and

Table 7.3: TCP parameters for TCP_{fast}

Parameter	Value
retransmission timer	0.01 seconds
fast retransmit threshold	one duplicate ack

the background sources are much more agressive. During a simulation with TCP_{fast}, the total number of retransmitted packets for all the sources is roughly three times larger than with $TCP_{default}$. This explains why TCP_{fast} achieves a throughput slightly lower than $TCP_{default}$ in our LAN environment. While on average the retransmission timer expired slightly less than four times during each 10 MBytes transfer on the privileged source with $TCP_{default}$ and a 180000 cells per second MCR, it expired on average almost 10 times during each 10 MBytes transfer with TCP_{fast}. Although the expirations of the retransmission timer with TCP_{fast} do not impose long idle times, they force the privileged source to frequently perform slow-start and congestion avoidance (figure 7.9). Due to these frequent expirations of the retransmission timer, the average congestion window size used by the privileged source is too small to allow it to efficiently utilize its minimum guaranteed bandwidth. Furthermore, with TCP_{fast} the privileged source retransmitted on average about 500 KBytes per 10 MBytes transfer, while $TCP_{default}$ retransmitted only about 200 KBytes. Thus, TCP_{fast} is not necessarily a better solution than $TCP_{default}$

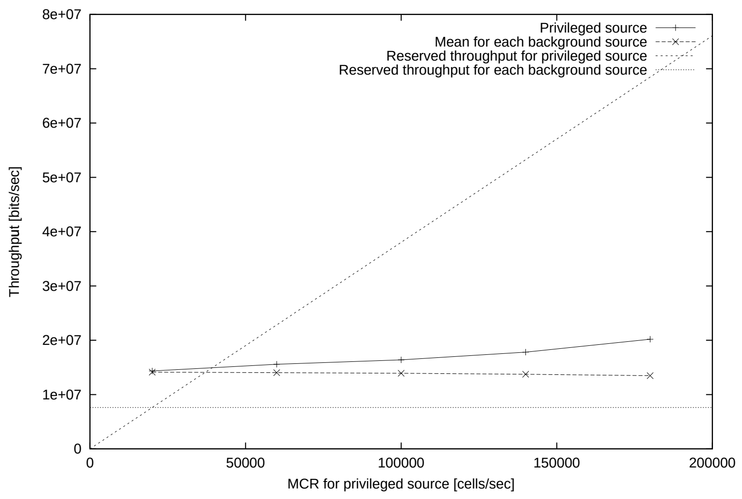

Figure 7.8: TCP_{fast} throughput for privileged and background sources

A third solution to improve the performance of TCP when packet losses occur would be to use the recently (re)proposed Selective Acknowledgements [MMFR96]. To evaluate the impact of this proposed TCP extension, we have patched the TCP code used in the STCP simulator with a SACK implementation [Mah96]. This TCP_{SACK} implementation[1]

[1]A brief description of the characteristics of this implementation may be found in appendix C.

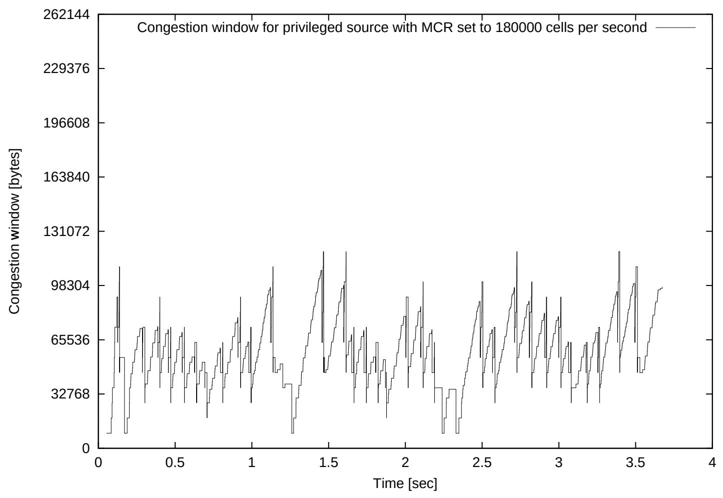

Figure 7.9: TCP_{fast} : sample congestion window trace

is rather conservative in its handling of retransmissions and follows [FF96]. Besides the use of the selective acknowledgements, TCP_{SACK} uses the same parameters as $TCP_{default}$ (table 7.4).

Table 7.4: TCP parameters for TCP_{SACK}

Parameter	Value
retransmission timer	0.2 seconds
fast retransmit threshold	3 duplicate acks
Selective Acknowledgements	Enabled

The simulations performed with TCP_{SACK} (figure 7.10) show that it achieves a better throughput than $TCP_{default}$ and TCP_{fast}. But TCP_{SACK} is still not able to efficiently use the guaranteed minimum bandwidth of the underlying ATM VC. A look at the simulation traces with a 180000 cells per second MCR for the privileged source showed that the percentage of CLP=1 AAL5-PDUs with TCP_{SACK} was similar to the percentage of CLP=1 AAL5-PDUs with $TCP_{default}$, but that the number of expirations of the retransmission timer was much lower (about one expiration per 10 MBytes transfer on average).

With TCP_{SACK}, the large granularity of the retransmission timer is not the only cause for the low TCP performance. Here, the low TCP performance is due to several related factors. First (and foremost), TCP is not able to adapt its behaviour to the F-GCRA. This causes the percentage of tagged AAL5-PDUs to be much larger than what could be expected if TCP was able to transmit at exactly its fairshare. Second, the output buffer at the bottleneck switch is heavily used, and its mean occupancy is close to 5000 cells for the whole simulation. This large output buffer occupancy corresponds to a relatively large round-trip-time for the sources (for example, with a 180000 cells per second MCR, the average packet round-trip-time for the privileged source was slightly larger than 14 milliseconds). To use its reserved throughput with such a round-trip-time, the privileged

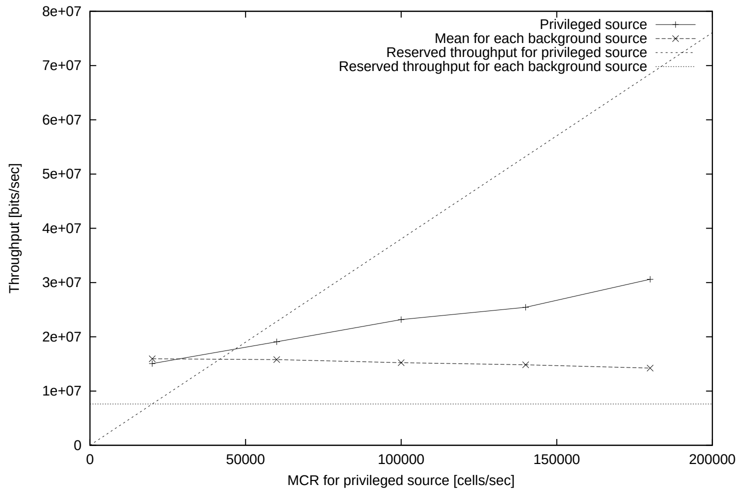

Figure 7.10: TCP_{SACK} throughput for privileged and background sources

source would need a window of at least 120 KBytes. Unfortunately, the large number of packet losses combined with TCP's congestion control algorithm force the congestion window of the privileged source to be much lower than this required value on average (see figure 7.11 for a sample trace of the congestion window with TCP_{SACK}).

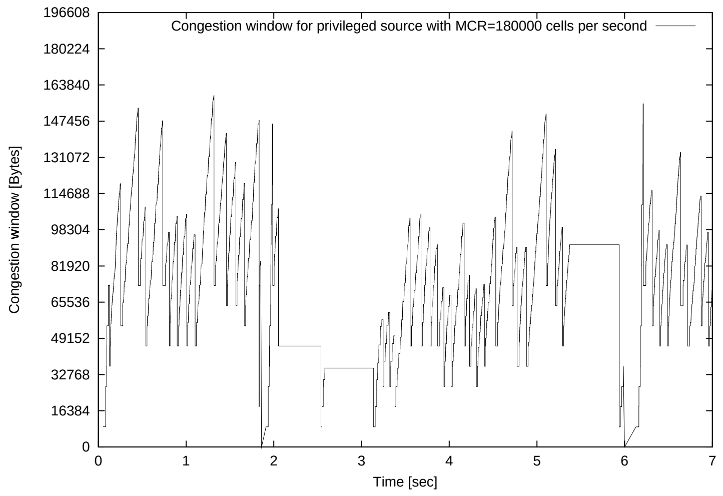

Figure 7.11: TCP_{SACK} : Sample congestion window trace

Influence of the background load

In the previous sections, we have shown the throughput achieved by the privileged source with nine background sources with a MCR set to 20000 cells per second. To verify that this particular value was not the reason for the low performance of TCP, we also performed simulations [Bon97] with nine background sources, but with their MCR set to 10000 and 5000 cells per second. These simulations produced similar results to those discussed in the previous sections. We even performed simulations with 9 background sources with a MCR of 0 cell per second (i.e. all the AAL5-PDUs sent by these sources are tagged). In this case, the privileged source had the same difficulties to use its reserved throughput as when the background sources used a non-zero MCR.

Influence of the number of background sources

Most of the simulations reported in this section were done with nine background sources. To verify that this particular number of background sources was not the cause of the low throughput achieved by the privileged source, we performed simulations with 3 and 6 background sources. In both cases, we set the sum of the MCRs of the background sources to 180000 cells per second. Figure 7.12 shows that even with three background sources, the privileged source still cannot use its reserved bandwidth with $TCP_{default}$. Simulations performed with TCP_{fast} and TCP_{SACK} produced similar results.

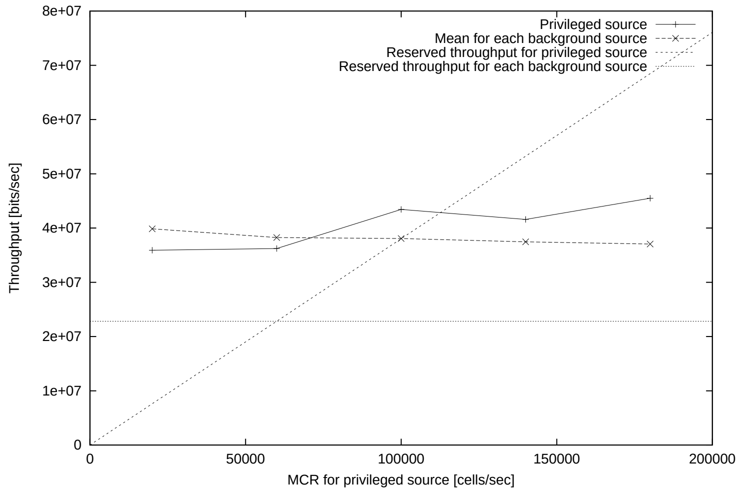

Figure 7.12: $TCP_{default}$: 3 background sources

Influence of the LBO threshold

Most of the simulations shown in this section were performed with output queues of 8192 cells and HBO and LBO thresholds of respectively 7168 and 6144 cells on the ATM switches. With these thresholds, no CLP=0 AAL5-PDUs were discarded in the bottleneck switch. Simulations performed with the LBO thresold set to 4096 and 2048 cells produced similar results as those with the LBO threshold set to 6144 cells. With TCP_{fast}, setting the LBO threshold to 2048 cells allowed the privileged source to attain a slightly higher throughput, but still less than 50% of its reserved throughput. However, this should not suggest the use of a LBO threshold set to 0 cell. In this case, all the CLP=1 AAL5-PDUs are discarded at the bottleneck switch. Simulations performed with this value of the LBO threshold show that, in this case, the throughput achieved by all the sources collapses. With $TCP_{default}$ and TCP_{SACK}, each TCP source is only able to achieve a throughput of 275 kilobits per second with their MCR set to 20000 cells per second. A look at the segment traces revealed that TCP successfully sends three 9140 bytes segments every 800 millliseconds. With its MCR set to 180000 cells per second, the privileged source still achieves only 275 kilobits per second. With TCP_{fast}, the situation if slightly better but far from satisfactory. TCP_{fast} successfully sends three 9140 bytes segments every 40 milliseconds. This collapse of the TCP throughput is due to the fact that TCP is not able to adapt its behaviour to a traffic contract enforced by a F-GCRA. Similar problems with the VBR GCRA were already discussed in chapter 6.

On the other side, it should also be noted that using a LBO threshold larger than the sum of the window sizes[2] used by the TCP sources is not a solution either. In this case, there are no losses in the switch buffers, and thus the AAL5-PDUs sent by all the sources are served independently of their CLP bit, and the throughput is shared fairly among all the competing sources, and thus the TCP throughput is independent of the MCR.

Influence of the TCP Maximum Segment Size

All the simulations presented in the previous sections were performed with a TCP Maximum Segment Size (MSS) of 9140 bytes. This value corresponds to the default value for Classical IP over ATM [Atk94]. However, several papers [RF95] [BKD96a] have shown that the TCP performance in an ATM network may also depend on the value of the MSS. To verify that the large value of the default TCP MSS used for our simulations was not the cause of the TCP performance problems, we performed new simulations with smaller values for the TCP MSS of all the sources (but with the same GFR traffic contracts as in the previous sections).

The throughput achieved by $TCP_{default}$ with a 512 bytes MSS is slightly lower than the throughput achieved with a 9140 bytes MSS (figure 7.6). The throughputs achieved by $TCP_{default}$ with the MSS sizes corresponding to an Emulated Ethernet and an Emulated Token Ring are between these two values. The 512 bytes MSS corresponds to the default MSS size used by TCP implementations which do not use Path MTU Discovery when they communicate with a destination which is in a different IP subnet. The fact that the throughput achieved by $TCP_{default}$ is almost independent of the MSS is not surprising since with $TCP_{default}$ the main limitation for the throughput is the large granularity of the retransmission timer.

The throughput achieved by TCP_{SACK} with the 1460 bytes MSS (figure 7.13) corresponding to an Emulated Ethernet is lower than the throughput achieved with the 9140 bytes MSS (figure 7.10). This is not surprising, as with TCP_{SACK} one of the reasons for the low throughput for the privileged source is the time spent with a small congestion window during congestion avoidance. During congestion avoidance, the congestion window is increased by one MSS-sized segment every round trip time, and thus a lower MSS means that the increase of the congestion window is slower.

Effect of heterogeneous MSS sizes

For the simulations discussed in the previous sections, all the sources used the same MSS. However, a real network may not be so homogeneous. For example, LAN Emulation

[2]This is the reason why we used of window size of 192 KBytes for our simulations to ensure that AAL5-PDUs are lost in the ATM switches.

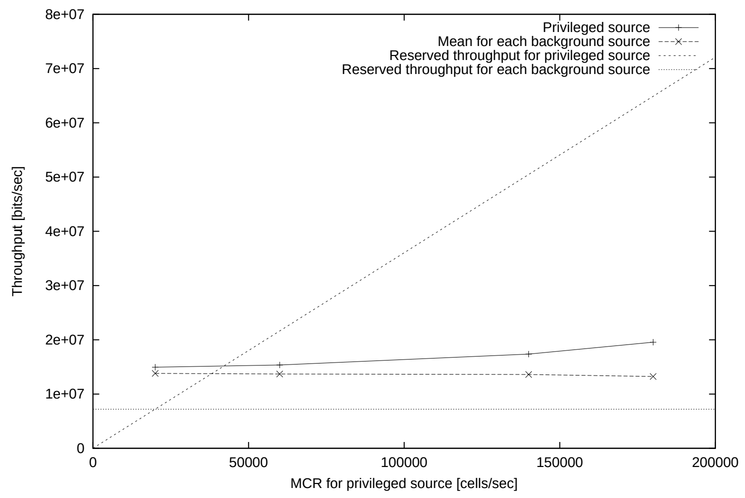

Figure 7.13: TCP_{SACK} : 1460 bytes MSS

supports four different maximum packet sizes. Sources which are part of different types of Emulated LANs may travel the same bottleneck link, and thus it is important to study how TCP behaves with the Double-EPD switch implementation when the privileged and the nine background sources do not use the same MSS[3].

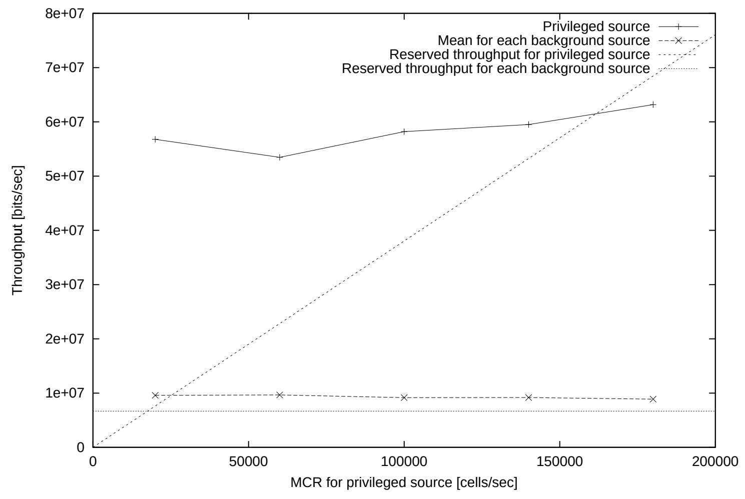

Figure 7.14: $TCP_{default}$ throughput with 9140 bytes MSS for privileged source and 512 bytes MSS for background sources

In figures 7.14 and 7.15, we report simulations performed with heterogeneous MSS sizes with $TCP_{default}$. Figure 7.14 shows the throughput achieved by the privileged and the background sources when the privileged source uses a 9140 bytes MSS while the background sources use a 512 bytes MSS. Figure 7.15 shows the throughput achieved when

[3]However, it should be noted that all GFR traffic contracts use the same MFS (200 cells) and MBS during all the simulations

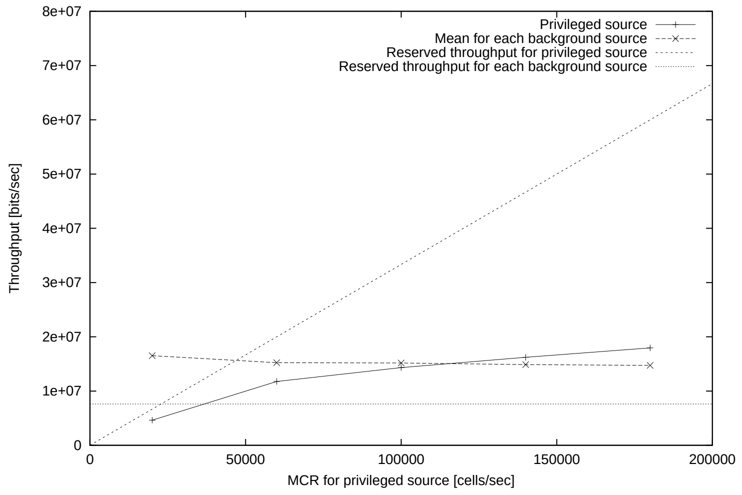

Figure 7.15: $TCP_{default}$ throughput with 512 bytes MSS for privileged source and 9140 bytes MSS for background sources

the privileged source uses a 512 bytes MSS, while the background sources use a 9140 bytes MSS. When the privileged source uses a much larger MSS than the background sources, it acquires a large share of the bottleneck link, and its throughput is almost independent of its MCR. On the opposite, when the privileged source uses a much smaller MSS than the background sources, its throughput is very low with a low MCR, and even with a 180000 cells per second MCR, it has huge difficulties to achieve a higher throughput than the background sources.

This unfairness for the sources which use a low MSS is again mainly due to the fact that the increase of the congestion window during congestion avoidance is proportional to the MSS size. For example, the simulations performed with a MCR of 180000 cells per second and a 512 bytes MSS for the privileged source revealed the following behaviour. Less than 1% of the segments sent by the privileged source were CLP=1 segments, and the retransmission timer expired on average twice per 10 MBytes transfer. These expirations of the retransmission timer are responsible for a fraction of the throughput drop, but the main cause for the throughput drop is the congestion avoidance mechanism. Most of the 10 MBytes transfer occured as follows. The privileged source performs slow-start, and several CLP=1 segments sent during this phase are discarded by the bottleneck switch. After the expiration of the retransmission timer, the privileged source retransmits these segments and performs congestion avoidance. Due to the small value of the MSS, the congestion window increases slowly. Unfortunately, the output buffer of the bottleneck switch is on average always more than 50% full. Such a high occupancy for the output buffer of the bottleneck switch corresponds to an average round-trip-time of 12.6 milliseconds. Combined with such a large round-trip-time, the low average value of the congestion window for the privileged source explains its low throughput. Similar unfairness occurs with $TCP_{default}$ and TCP_{fast} when the privileged source is part of an Emulated Token Ring and the background sources are part of an Emulated Ethernet or the reverse.

It should be noted that this unfair advantage for TCP sources with a large packet size over TCP sources with a smaller packet size also occurs with the UBR service category when the EPD buffer acceptance algorithm is used.

7.3.3 WAN simulations with the Double-EPD implementation

From the low performance of TCP in a LAN with the proposed Double-EPD implementation, there is little hope that TCP will achieve a better performance in a wide area network. Figure 7.16 presents the throughput achieved by TCP_{SACK} with a 9140 bytes MSS in a wide area network (UNI1 and UNI2 links have a delay of 2.5 msec, while the NNI link has a delay of 10 msec) whose switches uses 8192 cells wide output buffers with LBO and HBO thresholds set to respectively 6144 and 7168 cells. This simulation shows clearly that TCP is not able to benefit from the GFR service category with the proposed Double-EPD switch implementation in a WAN environment.

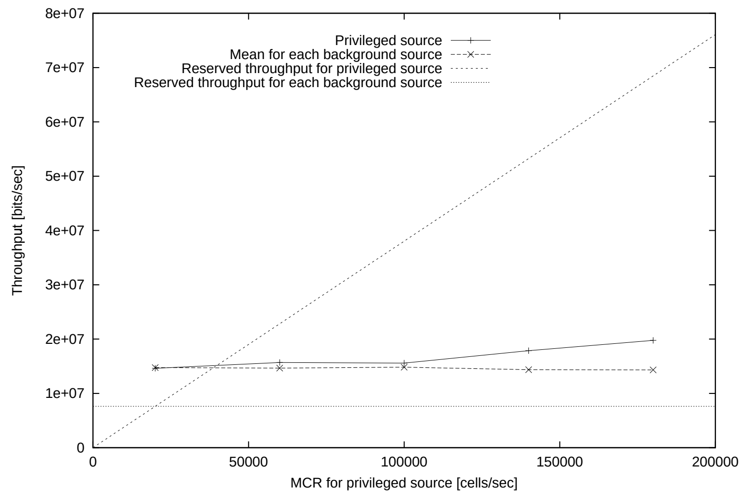

Figure 7.16: FIFO-based switches : TCP_{SACK} throughput in WAN

During the simulations with TCP_{SACK}, there were few expirations of the retransmission timer, and the main reason for the low throughput of the privileged source was the low average value of its congestion window due to the congestion avoidance mechanism.

7.3.4 LAN simulations with per-VC threshold and scheduling

To validate the Virtual Spacing scheduler used in the simulator, we performed several simulations in the ATM LAN[4] of figure 7.3 with a small TCP window (32 KBytes) so that the 8192 cells long output buffer of the ATM switches does not overflow. We used nine background sources for these simulations, and each background source had its MCR set to 20000 cells per second. As there are no cell losses in this environment, $TCP_{default}$, TCP_{fast} and TCP_{SACK} have identical performance. As expected, the simulations performed with

[4]For the simulations with the per-VC threshold and scheduling switch implementation discussed in sections 7.3.4 and 7.3.5, we considered only the GFR.1 conformance definition and assumed that the sources were only sending CLP=0 AAL5-PDUs. This is the most natural utilisation of the GFR.1 conformance definition with TCP/IP traffic.

a 9140 bytes MSS showed that there were no cell losses, and that the available throughput was fairly shared among the background and the privileged sources (figure 7.17).

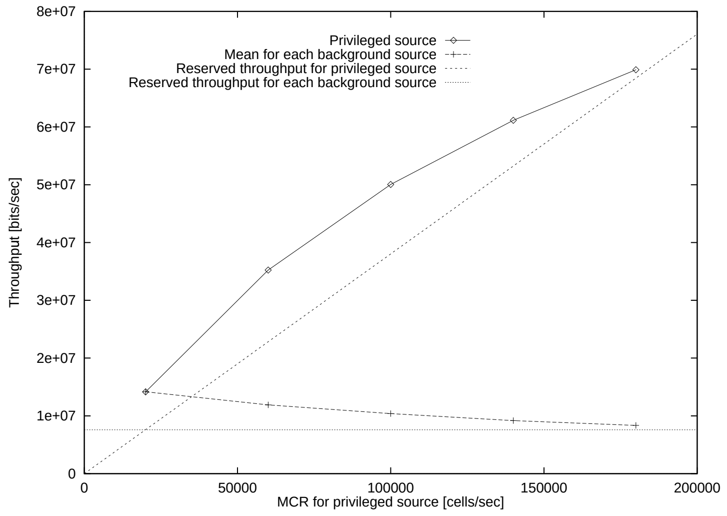

Figure 7.17: WFQ-based switches : *TCP* throughput for privileged and background sources

This simulation shows that with the proposed WFQ-based implementation in our simple LAN environment, the unreserved bandwidth is allocated in proportion to the MCR of the sources. This implies that a source with a much larger MCR than the other sources will achieve a much higher throughput, and may even use almost all the unreserved bandwidth. For example, if we consider the same ATM LAN but where the MCR of each background source is set to 5000 cells per second, the simulations show that the privileged source acquires a large proportion of the NNI link (figure 7.18), and this kind of distribution of the available bandwidth may not be desireable in every environment[5]. Furthermore, the delay experienced by the background sources may become very large. For example, with the MCR of the privileged source set to 180000 cells per second, the average packet delay from the privileged source to the privileged destination is equal to 1.16 milliseconds, while the average packet delay from a background source to the corresponding background destination is 32.7 milliseconds. While the GFR service category is not expected to provide a fair treatment in terms of transmission delay [GH97] a shorter delay for the background sources would probably be desireable.

LAN simulations with limited buffers

The simulations discussed in the previous section showed that TCP performed well with the proposed per-VC threshold and scheduling switch implementation when there were no packet losses. This is an important difference with the FIFO-based implementation as when there are no losses, the FIFO-based implementation does not allow the privileged source to use its reserved bandwidth. Unfortunately, in a real network, packet losses will probably occur due to the limited size of the switch buffers. In our simple LAN, with

[5]This potential problem was already mentioned in [Bon97].

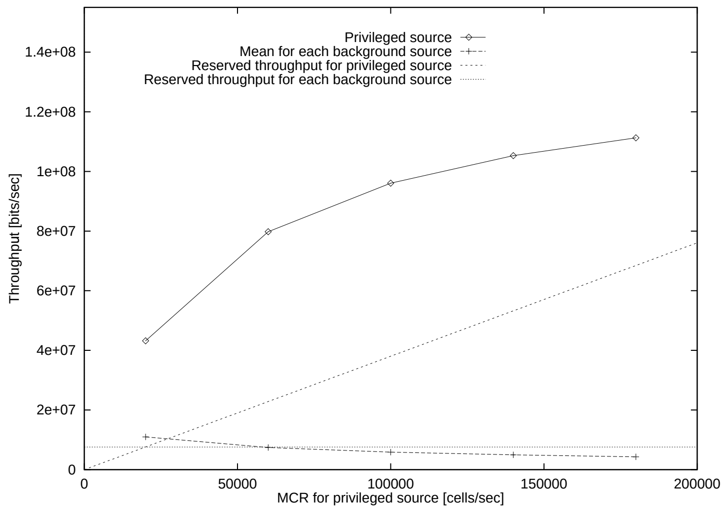

Figure 7.18: TCP throughput with $\sum MCR_{background} = 9 \times 5000$ cells/sec

a buffer of 8192 cells, packet losses will occur when the TCP sources use a 64 KBytes window.

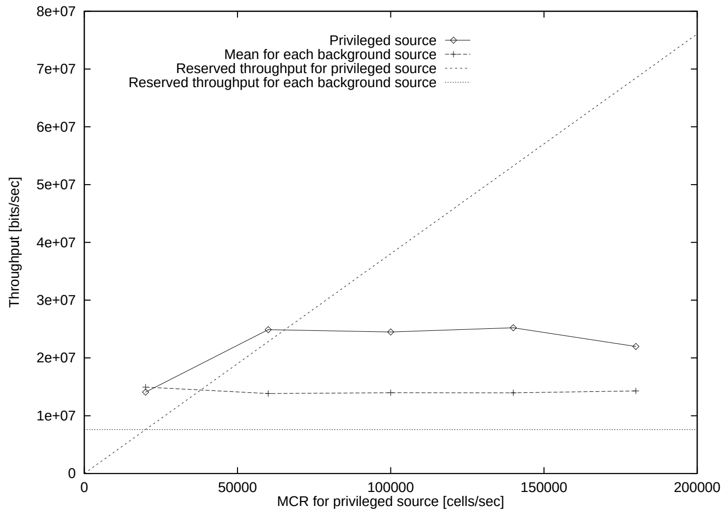

Figure 7.19: TCP_{SACK} with 64 KBytes window and 9140 bytes MSS in LAN

With a 9140 bytes MSS, a 64 KBytes window and TCP_{SACK}, the privileged source had a lot of difficulties to efficiently use its reserved throughput (figure 7.19). However, with a 1460 bytes MSS it was almost able to achieve its reserved throughput (figure 7.20).

The large difference between the 1460 bytes and the 9140 bytes MSS is due to the fact that the retransmission and congestion control mechanisms are closely related in TCP_{SACK}. With a 9140 bytes MSS, a maximum window size of 65535 bytes corresponds to 7 segments. When a packet loss is detected by TCP_{SACK}, it performs congestion avoidance and thus its congestion window is reduced by a factor of two. If a new segment is lost, this loss will only be taken into account by the TCP_{SACK} sender when it has received three duplicates acknowledgements. Thus, if the number of lost segments is

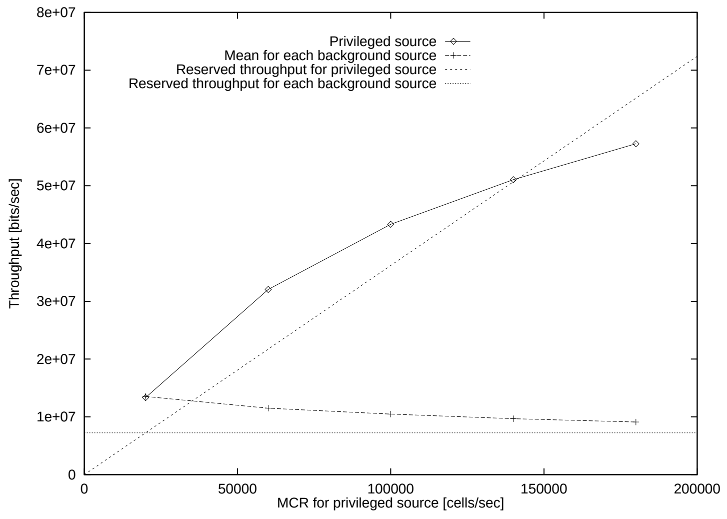

Figure 7.20: TCP_{SACK} with 64 KBytes window and 1460 bytes MSS in LAN

larger than the difference between the current value of the congestion window and the retransmit threshold (3 with TCP_{SACK}), the TCP_{SACK} sender will be forced to wait for the expiration of the retransmission timeout. The packet traces gathered during the simulations revealed that for the privileged source, 50% of the segment losses occured in groups of at least two segments, with a sequence gap of less than 65535 bytes between the lost segments. With a 9140 bytes MSS, the retransmission timer expired on average 10 times per 10 MBytes transfer, while with the 1460 bytes MSS, it expired on average only once per 10 MBytes transfer. This explains the large difference between the 9140 and the 1460 bytes MSS.

Simulations performed with TCP_{SACK} and heterogenous MSS sizes in a LAN environment showed than the per-VC threshold and scheduling implementation achieved a better fairness that the proposed Double-EPD implementation. With TCP_{SACK}, there were almost no difference when the privileged source used a 512 bytes MSS while the background sources used a 9140 bytes MSS or the opposite [Bon97]. In both cases, the privileged source was able to achieve almost its weighted fair throughput.

7.3.5 WAN simulations with the per-VC threshold and scheduling implementation

To evaluate the performance of TCP in a wide area network, we performed new simulations with the same network topology as in figure 7.3 , but with the delay on the NNI link set to 10 milliseconds and the delay on the UNI links set to 2.5 milliseconds. Thus, without taking into account the queueing delays, the round trip time in this network is 30 milliseconds. To fully utilize the link, the TCP sources should use a window size at least equal to the bandwidth delay product. The simulations performed with a slightly larger window size (550 KBytes) showed that with TCP_{SACK} the privileged source was not able to utilize its reserved throughput (figure 7.21), although the throughput of the privileged source was much higher than with the FIFO-based implementation. The main reason for the low TCP throughput is again the long time spent by the privileged TCP source in congestion avoidance phase with a congestion window which is smaller than $MCR \times rtt$. This performance problem could probably be partially solved by modifying TCP in a similar way as proposed in [FKSS97] for the controlled load service in an integrated services Internet. This article proposed two modifications to the TCP implementations to better utilize the controlled load service. The first modification is to change the congestion avoidance mechanism so that the congestion window does not become smaller than the *reserved bandwidth* $\times$ *rtt* product. This ensures that the congestion window will always be large enough to utilize the reserved bandwidth. The second modification is to utlize a timer-triggered transmission mechanism. In a standard TCP source, the transmission of a new segment is usually triggered by the reception of an acknowledgment, provided that this segment fits into its congestion window. With the second modification, a TCP source may send a new TCP segment upon expiration of the transmission timer, even if this segment is outside the congestion window. This modification allows the TCP source to better utilize its reserved bandwidth even when the returning acknowledgments are slightly delayed.

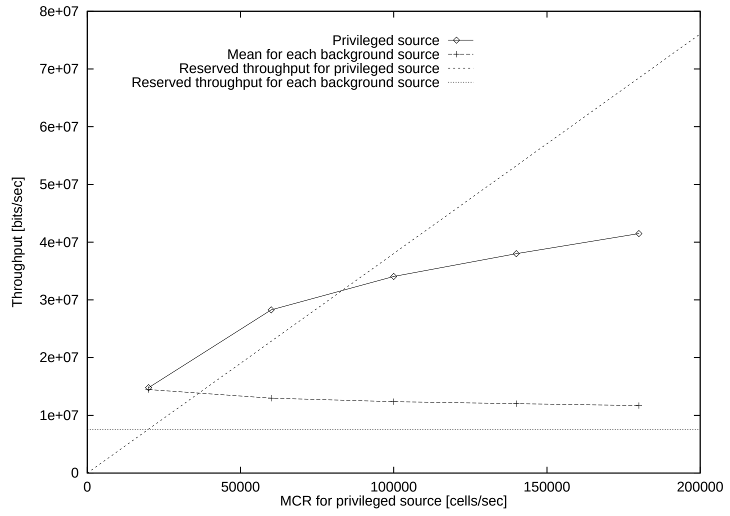

Figure 7.21: TCP_{SACK} throughput with 550 KBytes window in a WAN

7.3.6 Impact of tagging on the TCP performance with the per-VC threshold and scheduling implementation

In the previous sections, we have discussed the performance of TCP with the per-VC threshold and scheduling switch implementation used with the GFR.1 conformance definition. In this section, we evaluate the performance of TCP with the same implementation used with the GFR.2 conformance definition. Thus, we inserted an F-GCRA on the UNI links to tag the AAL5-PDUs which are not eligible for the bandwidth guarantee.

LAN Simulations

Simulations performed with 64 KBytes windows, TCP_{SACK} and homogeneous MSS sizes (9140 or 1460 bytes) for the privileged and the background sources produced similar results as those presented in section 7.3.4. Thus, in this case, the tagging performed by the F-GCRA did not seem to have an influence on the throughput of the privileged and background sources.

Simulations performed with heterogeneous MSS sizes show that unfairness may occur, but the unfairness differs from the unfairness discussed with the Double-EPD implementation. Figure 7.22 shows the throughput achieved by TCP_{SACK} in a LAN when the MSS size of the privileged source is set to 9140 bytes, while the background sources use a 512 bytes MSS size. In this case, there is no significant unfairness. However, the simulations show that when the privileged source uses a 512 bytes MSS size, while the background sources use a 9140 bytes MSS size, a large unfairness occurs (figure 7.23). In this case, the lower throughput for the privileged source is mainly due to the large number of expirations of its retransmission timer. This is in contrast with the simulations performed with the proposed Double-EPD implementation (figure 7.15) where the number of expirations of the retransmission timer of the privileged source was very low and the low throughput was caused by the slow increase of the congestion window. With a MCR set to 20000 cells per second, the retransmission timer of the privileged source expired more than 20 times per 10 MBytes transfer during the simulations with the WFQ-based implementation reported in figure 7.23 while it expired only once per 10 MBytes transfer during the simulations with $TCP_{default}$ and the Double-EPD implementation reported in figure 7.15. Similar unfairness occured with TCP_{fast}.

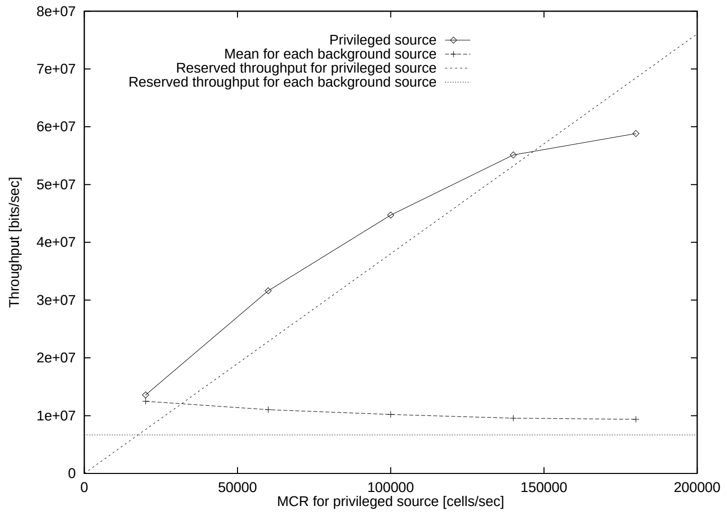

Figure 7.22: TCP_{SACK} throughput with 64 KBytes window and 9140 bytes MSS for privileged source and 512 bytes MSS for background sources

WAN Simulations

In a WAN, the impact of tagging on the TCP throughput is much higher than in a LAN. The simulations performed with a 550 KBytes window size showed that with TCP_{SACK} the privileged source was not able to utilize its reserved throughput (figure 7.24). It should be noted that when tagging is used at the network access point, the TCP_{SACK}

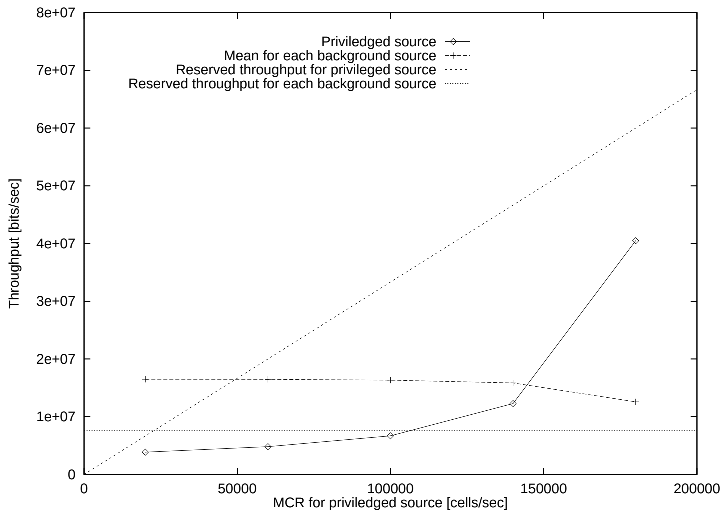

Figure 7.23: TCP_{SACK} throughput with 64 KBytes and 512 bytes MSS for privileged source and 9140 bytes MSS for background sources

throughput of the privileged source is much lower in a WAN than when no tagging is used.

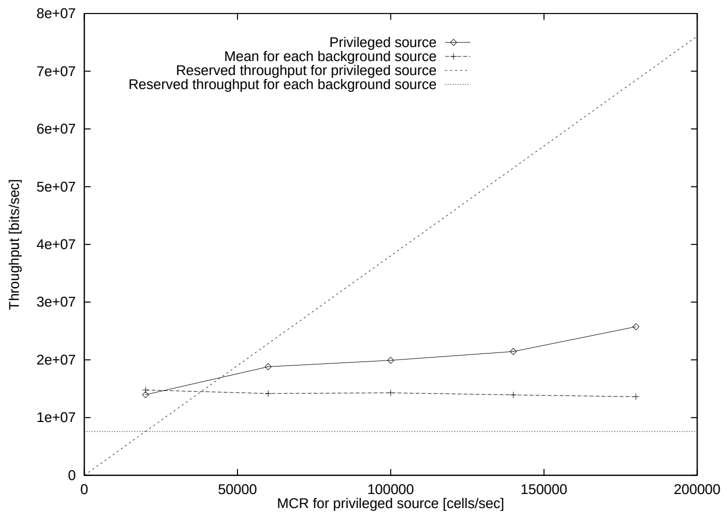

Figure 7.24: TCP_{SACK} throughput with 550 KBytes window in WAN

The main reason for the low performance of TCP lies in how TCP is able to adapt its rate to a traffic contract enforced by the F-GCRA. Let us for example consider the WAN simulation with TCP_{SACK} shown in figure 7.24. When the MCR of the privileged source is set to 180000 cells per second, while the MCR of each background source is set to 20000 cells per second, the privileged source should only transmit at a rate slightly above 180000 cells per second on average as this is the rate it should receive from the WFQ scheduler on the bottleneck switch. The simulations show that this does not happen. In fact, the measurements of the arrival rate (averaged during a period of 100 milliseconds) of the privileged source at the bottleneck switch show that the privileged source rarely reaches its reserved throughput during a 100 milliseconds period (figure 7.25). However, even if on average the privileged source does not utilize its reserved bandwidth its AAL5-PDU flow is too bursty for the F-GCRA. On average, more than one fifth of the AAL5-PDUs sent by the privileged source are tagged by the F-GCRA. This is mainly due to the fact that the congestion control scheme used by TCP forces the traffic to be bursty. During slow-start, the traffic is bursty because of the exponential increase of the congestion window. During congestion avoidance, the traffic is also bursty. When the congestion window is smaller than the bandwidth delay product, TCP sends a congestion window worth of segments, then is idle until the return of the acknowledgements.

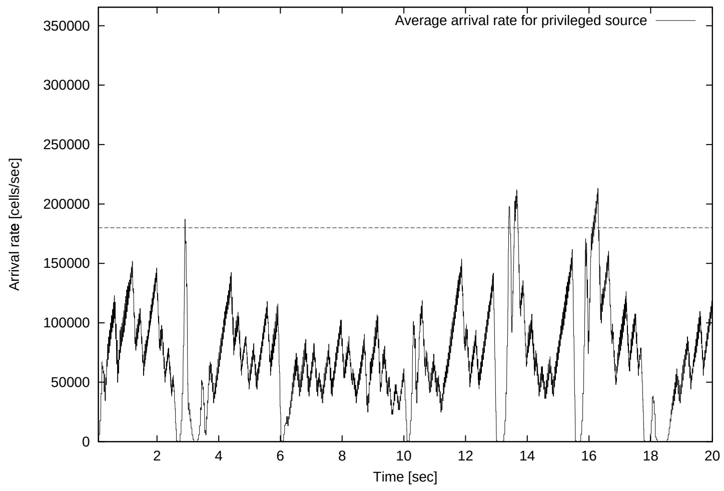

Figure 7.25: Arrival rate on bottleneck switch for privileged source

7.3.7 Related work

Our simulations with workstation traffic [Bon97] were carried shortly after the beginning of the work on GFR within the ATM Forum and were dsitributed within the ATM Forum and ITU-T in July 1997. Besides our work, several simulation studies of TCP with the GFR service category have been presented, mainly within the ATM Forum, since the introduction of the GFR service category [PB97, GJF$^+$97, HLK97, EA98, WC98b].

[PB97] discusses the performance of TCP in a LAN environment with a FIFO scheduler, a round-robin scheduler and a weighted round robin (WRR) scheduler. Their simulations show, like our simulations, that the FIFO scheduler and the round-robin scheduler are not sufficient, even when combined with a F-GCRA to support the GFR service guarantees. They note that with such schedulers, "*the way to ensure a minimum rate guarantee is with a combination of a large packet size and LBO close to zero*". Our simulations disagree with this conclusion. This difference between our simulations and those discussed in [PB97] is due to the fact that they use a 100 μsec granularity for the TCP retransmission timer in their simulation model. As already mentioned, we do not consider such a low granularity to be a valid model of current TCP implementations.

[GJF$^+$97] proposes another implementation for the GFR service category in FIFO switches and studies briefly the performance of TCP with this switch implementation.

However, [GJF$^+$97] does not compare the performance of the proposed implementation with other switch implementations.

[HLK97] studies the performance of the per-VC threshold and scheduling implementation with workstation traffic. This paper considers LAN and WAN environments and two variants of the per-VC threshold and scheduling implementation. The first variant is the per-VC threshold and scheduling implementation as described in section 7.2.2 with a Weighted Round-Robin (WRR) scheduler. The second variant also uses a WRR scheduler, but does not utilize per-VC accounting and a threshold for each VC. The simulations discussed in [HLK97] compare these two variant implementations with and without tagging performed by a F-GCRA at the ingress of the network. With the first variant implementation, the simulations show that the tagging does not have a significant influence on the performance of TCP. This is slightly different from our simulation results [Bon97], but is probably due to the low granularity for the retransmission timer (1 millisecond for the LAN simulations) and to the choice of larger values for the LBO and T_i thresholds in [HLK97] compared with our simulations. The simulations with the second variant implementation show that the performance of TCP decreases when tagging is used at the ingress of the network. Furthermore, [HLK97] also shows that TCP can suffer from a huge unfairness when it has to compete with a greedy source with the second variant implementation.

[EA98] proposes three alternative switch implementations which support the GFR.2 conformance definition in FIFO-based switches and studies their performance with workstation traffic. The first implementation is a modification of the Double-EPD implementation inspired by the drop-from front packet discard strategy proposed in [LNO96]. In this implementation, when a new CLP=1 AAL5-PDU arrives from VC[i] and the buffer occupancy is above the LBO threshold, then the state of the switch is modified so that the first complete CLP=1 AAL5-PDU from VC[i] to be transmitted on the output link will be discarded.

The second implementation is also a modification of the Double-EPD implementation which requires per-VC accounting and [EA98] proposes to use two counters for each VC to keep track of the number of both CLP=0 (resp. CLP=1) cells in the buffer. This implementation also uses two global counters to keep track of the total number of CLP=0 (resp. CLP=1) cells in the buffer in addition to the total buffer occupancy. When a new CLP=1 AAL5-PDU arrives, it is accepted if either the total buffer occupancy is below the LBO threshold or the CLP=1 cells from this VC are using less than their fairshare of the buffer. The CLP=1 cells from a VC are using less than their fairshare of the buffer if $Q_{CLP1}[i] \leq Q_{CLP1}/Na$ where $Q_{CLP1}[i]$ is the number of CLP=1 cells from VC[i] in the buffer, Q_{CLP1} is the total number of CLP=1 cells in the buffer and Na is the number of active VCs (a VC is considered to be active when at least one cell from the VC is waiting in the buffer). This implies that the second implementation proposed in [EA98] tries to distribute the available buffer space for CLP=1 cells equally among all the active VCs. When a new CLP=0 AAL5-PDU arrives, it is accepted if the total buffer occupancy is below the HBO threshold *or* if the CLP=0 cells from this buffer are using less than their fairshare of the buffer. We are rather sceptical concerning the usefulness of this last test. When the GFR.2 conformance definition is used, CLP=0 AAL5-PDUs should only be discarded when a severe congestion occurs (e.g. buffer is full). We consider that the CLP=0 cells should *only* be discarded if the total buffer occupancy is above the HBO threshold.

The third implementation proposed in [EA98] combines the features of the other implementations. The simulations discussed in [EA98] study the performance of two TCP implementations with enhanced retransmission mechanisms in a LAN environment. The two studied TCP implementations are TCP_{SACK} as we studied in the previous sections and TCP NewReno [Hoe96]. The simulations performed with the Double-EPD implementation confirm the simulations discussed in section 7.3.2. [EA98] then studies the impact of the MFS and MBS parts of the GFR traffic contract and the LBO threshold on the performance of the two TCP implementations with Double-EPD and the three proposed switch implementations. The simulations show that the MFS should not be too large, the MBS should be larger than twice the MFS to achieve good performance with workstation traffic and that the LBO threshold should not be too small.

7.4 Performance with internetwork traffic

For this simulation study, we have considered three simple scenarios : the "five routers", the "ten routers" and the "mixed ten routers" scenarios. These scenarios enable to study different aspects of the performance of internetwork traffic with the GFR service category. The "five routers" allows us to evaluate whether internetwork traffic can benefit from a guaranteed minimum bandwidth in the ATM network. With the "ten routers" scenario, we study the impact of the round-trip-time on the efficient utilization of the ATM VCs. For these two scenarios, we consider the Double-EPD and the per-VC threshold and scheduling implementations. Finally we use the "mixed ten routers" scenario to evaluate the impact of the GFR conformance definition on performance of internetwork traffic with the per-VC threshold and scheduling implementation.

In the "five routers" scenario[6] (figure 7.26), all the ATM links have a bandwidth of 34 Mbps. The delay on the link between the ATM switches is set to 10 milliseconds, while the delays on the links between the ATM switches and the routers is set to 2.5 milliseconds. The switches have per-port output buffers of 16.000 cells while the routers are tail-drop routers with 700 KBytes buffers. Twenty TCP sources are attached with 10 Mbps point-to-point links to each router on the left side. Each of these sources sends traffic to a companion destination attached to the companion router on the right side (i.e. a source attached to R3 sends traffic to a destination attached to R3*). Two companion routers (e.g. R1 and R1*) are linked with one GFR VC. The characteristics of the different VCs are summarized in table 7.5. The total amount of reserved bandwidth on the link between the two ATM switches is set to 30 Mbps (out of a maximum of 34 Mbps).

The "ten routers scenario" (figure 7.27) will mainly allow us to evaluate the influence of the round-trip-time on the performance of the two switch implementations. In this scenario, all the links between the routers and the ATM switches still have a bandwidth of 34 Mbps. The delay on the link between the two switches is set to 5 milliseconds. As in the "five routers" scenario, 20 TCP sources or destinations are attached to each router. We consider five different delays for the links between the routers and the ATM switches and GFR VCs with a MCR set to 2 or 4 Mbps. The characteristics of the GFR VCs

[6]The five routers scenario is similar to the scenario used in section 6.4.2 to evaluate the suitability of the VBR.3 conformance definition to efficiently support internetwork traffic.

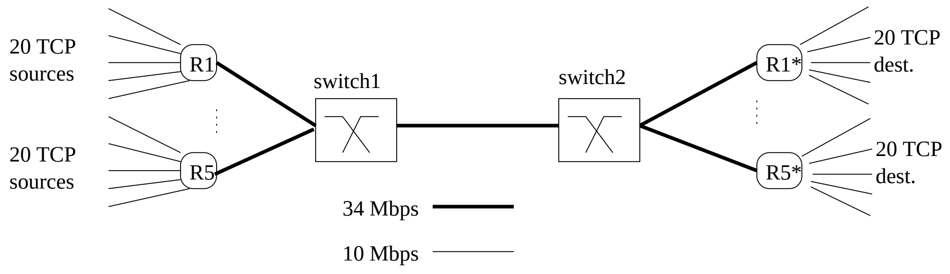

Figure 7.26: The five routers scenario

Table 7.5: Five routers scenario : characteristics of the GFR VCs

VC	PCR	MCR	Router-router delay
R1-R1*	34 Mbps	2 Mbps	15 msec
R2-R2*	34 Mbps	4 Mbps	15 msec
R3-R3*	34 Mbps	6 Mbps	15 msec
R4-R4*	34 Mbps	8 Mbps	15 msec
R5-R5*	34 Mbps	10 Mbps	15 msec

are summarised in table 7.6. In this scenario, the total reserved bandwidth on the link between the two switches is still 30 Mbps.

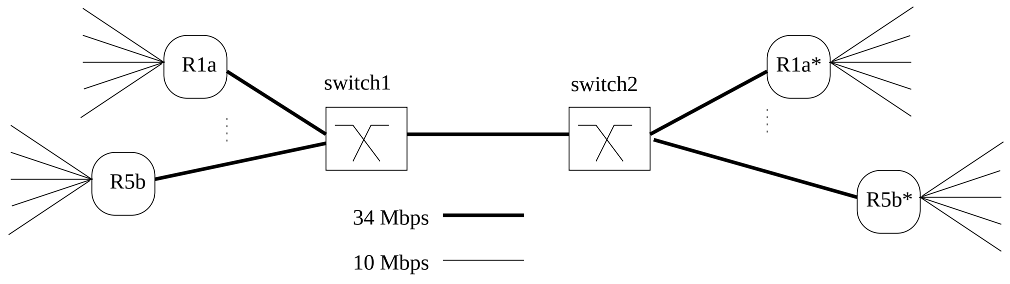

Figure 7.27: The ten routers scenario

In the "mixed ten routers" scenario (figure 7.28) , we modify the five routers scenario in order to compare the performance of the GFR.1 and GFR.2 conformance definitions with the per-VC threshold and scheduling implementation. For this, we consider five router-router VCs with the GFR.1 conformance definition and five router-router VCs with the GFR.2 conformance definition. We also double the bandwidth of the backbone ATM link. In the "mixed ten routers" scenario, the total reserved bandwidth on the backbone link is set to 60 Mbps out of 68 Mbps (2×80000 cells per second). The traffic contracts of the ten router-router VCs are summarized in table 7.7.

In the three scenarios, the TCP sources model greedy file servers. Each TCP source behaves as follows. It opens a TCP connection to its companion destination, sends a 1 MByte file and closes the TCP connection when the entire file has been successfully

Table 7.6: Ten routers scenario : characteristics of the GFR VCs

VC	PCR	MCR	Router-router delay
R1a-R1a*	34 Mbps	2 Mbps	25 msec
R1b-R1b*	34 Mbps	4 Mbps	25 msec
R2a-R2a*	34 Mbps	2 Mbps	45 msec
R2b-R2b*	34 Mbps	4 Mbps	45 msec
R3a-R3a*	34 Mbps	2 Mbps	65 msec
R3b-R3b*	34 Mbps	4 Mbps	65 msec
R4a-R4a*	34 Mbps	2 Mbps	85 msec
R4b-R4b*	34 Mbps	4 Mbps	85 msec
R5a-R5a*	34 Mbps	2 Mbps	105 msec
R5b-R5b*	34 Mbps	4 Mbps	105 msec

Table 7.7: Mixed ten routers scenario : characteristics of the GFR VCs

VC	PCR	MCR	Router-router delay	Conformance definition
R1a-R1a*	34 Mbps	2 Mbps	15 msec	GFR.1
R2a-R2a*	34 Mbps	4 Mbps	15 msec	GFR.1
R3a-R3a*	34 Mbps	6 Mbps	15 msec	GFR.1
R4a-R4a*	34 Mbps	8 Mbps	15 msec	GFR.1
R5a-R5a*	34 Mbps	10 Mbps	15 msec	GFR.1
R1b-R1b*	34 Mbps	2 Mbps	15 msec	GFR.2
R2b-R2b*	34 Mbps	4 Mbps	15 msec	GFR.2
R3b-R3b*	34 Mbps	6 Mbps	15 msec	GFR.2
R4b-R4b*	34 Mbps	8 Mbps	15 msec	GFR.2
R5b-R5b*	34 Mbps	10 Mbps	15 msec	GFR.2

transmitted. At that time, it opens a new TCP connection to the same destination and sends another 1 MBytes file ... The main parameters of the TCP sources are summarised in table 7.8. These values are similar to those found in commercially available TCP implementations.

The simulations discussed in this section were performed with STCP. Unless otherwise mentioned, all the simulations reported in this section correspond to an average over 200 seconds of simulated time.

While the GFR service category allows the endsystems (in our cases the routers) to distinguish between high and low priority AAL5-PDUs by marking the lower priority AAL5-PDUs, we assume in this section that all the routers send only CLP=0 AAL5-PDUs. For the simulations with the Double-EPD implementation, an F-GCRA is used at the UNI interfaces of the ATM switches to tag the AAL5-PDUs which are not eligible for the MCR. Thus, we always consider the GFR.2 conformance definition with the Double-EPD

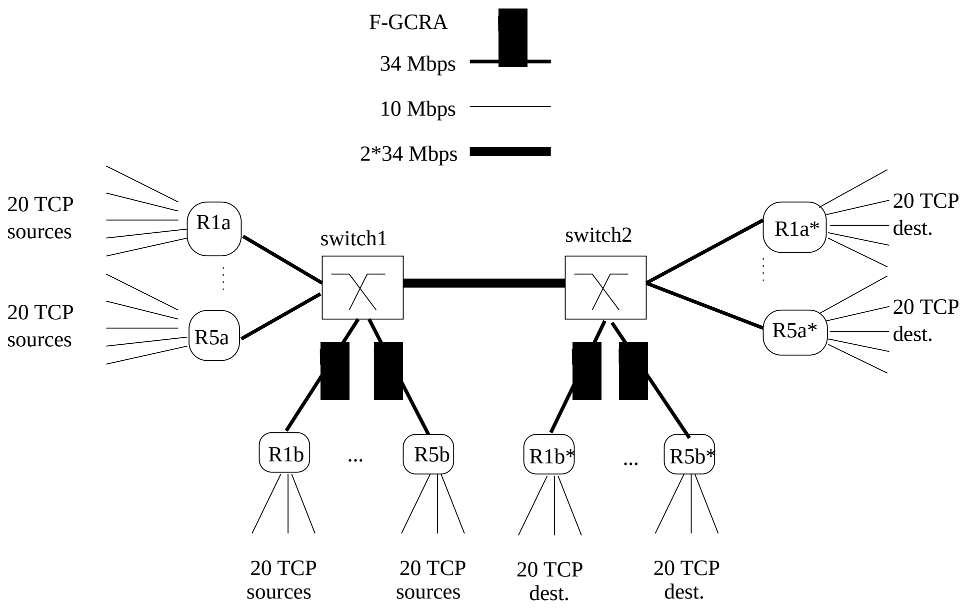

Figure 7.28: The "mixed ten routers" scenario

Table 7.8: Simulation parameters for the TCP sources

Parameter	Value
Window size	65535 bytes
Retransmission timer granularity	200 msec
Delayed Ack. timer granularity	50 msec
Maximum Segment Size	1460 bytes
File size	1 Megabyte

switch implementation. For the simulations with the per-VC threshold and scheduling implementation, we only consider the GFR.1 conformance definition with the five routers and ten routers scenario since this switch implementation was designed with GFR.1 in mind.

7.4.1 The five routers scenario

Before discussing the simulations with the GFR switch implementations, it is useful to first verify that the 20 Ethernet sources attached to each router are able to actually utilize the bandwidth reserved on the router-to-router VCs. For this, we slightly modify the five routers scenario. We assume that the five VCs use the UBR service category and use EPD on the ATM switches, but we artificially reduce the PCR of the links between switch2 and the destination routers as shown in table 7.9. The PCR of the ATM links connected to switch1 remains at 34 Mbps.

The simulations performed with this modified scenario (figure 7.29) showed that the 20 TCP sources were able to efficiently utilize the bandwidth available on the links between

Table 7.9: Five routers "UBR" scenario : links between switch2 and the routers

Link	PCR
switch2-R1*	2 Mbps
switch2-R2*	4 Mbps
switch2-R3*	6 Mbps
switch2-R4*	8 Mbps
switch2-R5*	10 Mbps

switch2 and the destination routers. In figure 7.29, we compare the total goodput of the 20 sources attached to each router with the theoretical optimum goodput. By optimum goodput, we mean the total goodput that the sources connected to each router would achieve if there were no losses and retransmissions. Our simulations showed that the total goodput achieved by the 20 sources was about 95% of the optimum goodput. In this section, we will use the total goodput obtained with the UBR service category as the basis to evaluate whether one GFR switch implementation allows the TCP sources to utilize at least the reserved bandwidth.

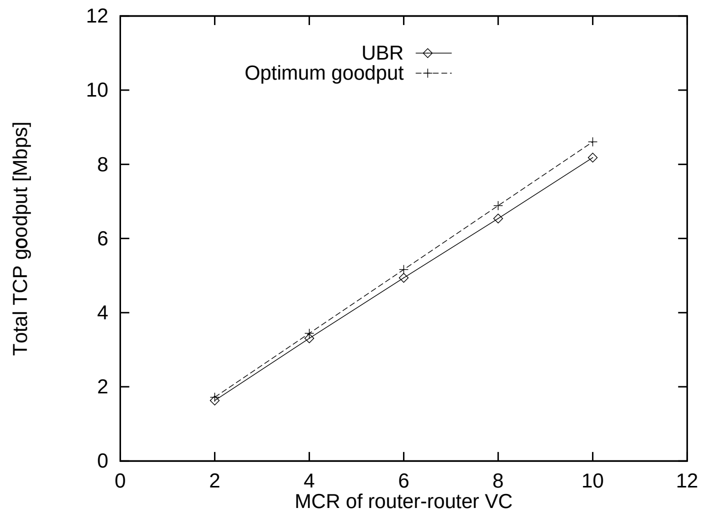

Figure 7.29: Five routers : UBR reference

Double-EPD

For the simulations performed with Double-EPD, we set the HBO threshold to 15000 cells. We set the LBO threshold to 2000 cells and varied the values of the MBS of all the GFR VCs. Our simulations with Double-EPD showed that the MBS has a significant influence on the total TCP goodput (figure 7.30). When the MBS was equal to the MFS, the sources attached to the 10 Mbps router could not utilize the reserved bandwidth of their router-to-router VC. On the other hand, the total goodput for the sources attached to the 2 Mbps router was more than twice the UBR reference goodput. When the MBS of all the GFR VCs was set to $20 \times MFS$, the sources attached to the 10 Mbps could utilize their reserved bandwidth.

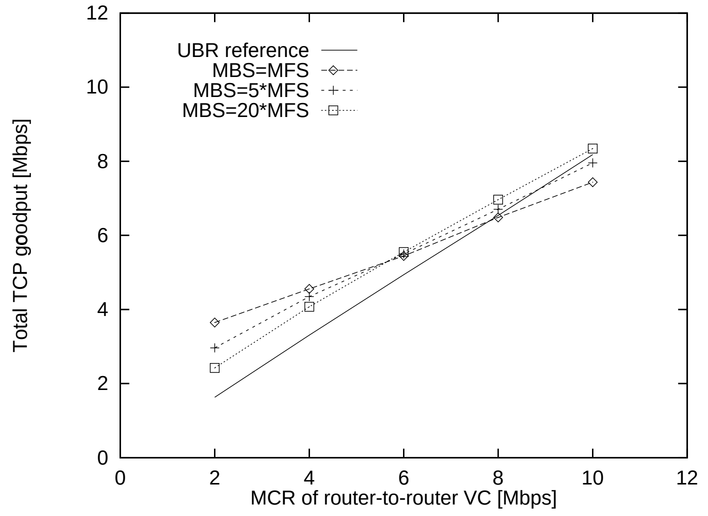

Figure 7.30: Five routers : Double EPD with LBO=2000

The main reason for the low performance with a small MBS is the large number of AAL5-PDUs which are tagged by the F-GCRA at the ingress of the network. For example, when MBS is set to MFS, 70% of the AAL5-PDUs sent by the 2 Mbps router and 30% of the AAL5-PDUs sent by the 10 Mbps router are tagged by the F-GCRA. Furthermore, although the sum of the MCR of the GFR VCs corresponds to 88% of the backbone link, only slightly less than 60% of the available bandwidth is used to carry CLP=0 AAL5-PDUs with MBS=MFS. With MBS set to $20 \times MFS$, 84% of the bandwidth on the backbone link is used to carry CLP=0 AAL5-PDUs.

An interesting point to mention is that during these simulations, the average buffer occupancy of the bottleneck switch was close to the LBO threshold (about 1700 cells with a MBS of 160 cells and a LBO threshold of 2000 cells). Furthermore, the maximum buffer occupancy did not reach very high values. For example, with the LBO threshold set to 2000 cells and a MBS of 640 cells, the buffer occupancy of the bottleneck switch did not became larger than 2600 cells.

Since the Double-EPD switch implementation is an AAL5-aware version of the PBS implementation used to support VBR.3 and the F-GCRA is an AAL5-aware version of the GCRA used with the VBR.3 conformance definition, it is interesting to compare the performance of internetwork traffic with VBR.3 and GFR.2. We compared the results of the simulations discussed in section 6.4.2 (figure 6.12) with the GFR.2 simulations discussed above. When we compare simulations with similar parameters (LBO=$PBS_{threshold}$ and MBS[VBR.3]=MBS[GFR.2]+MFS-1 since the F-GCRA accepts larger bursts than the VBR.3 GCRA), it appears that the Double-EPD implementation allows the sources to achieve a higher goodput than the PBS implementation. For example, when the LBO/PBS threshold is set to 2000 cells and the GFR.2 MBS is equal to MFS (32 cells), the sources attached to the 10 Mbps router achieve a total goodput of 7.4 Mbps with Double-EPD and GFR.2 while the same sources achieve only a total goodput of 6 Mbps with the PBS implementation and the VBR.3 conformance definition. In both cases, the fraction of the AAL5-PDUs tagged by the F-GCRA/GCRA is similar. The main reason why Double-EPD performs better than PBS is because it only discards entire AAL5-PDUs and thus partially corrupted AAL5-PDUs are not transmitted uselessly on the backbone link as with the PBS implementation combined with a VBR.3 GCRA.

We also performed simulations with other values for the LBO threshold. They showed that with a larger LBO threshold, the total goodput for the sources attached to the 10 Mbps router is slightly smaller than with a lower LBO threshold. On the other hand, they also showed that the total goodput for the sources attached to the 2 Mbps router increased slightly when the LBO threshold increased. For example, figure 7.31 compares the total goodput achieved by the sources attached to the various routers when MBS is equal to MFS with a LBO threshold set to 2000 and 12000 cells.

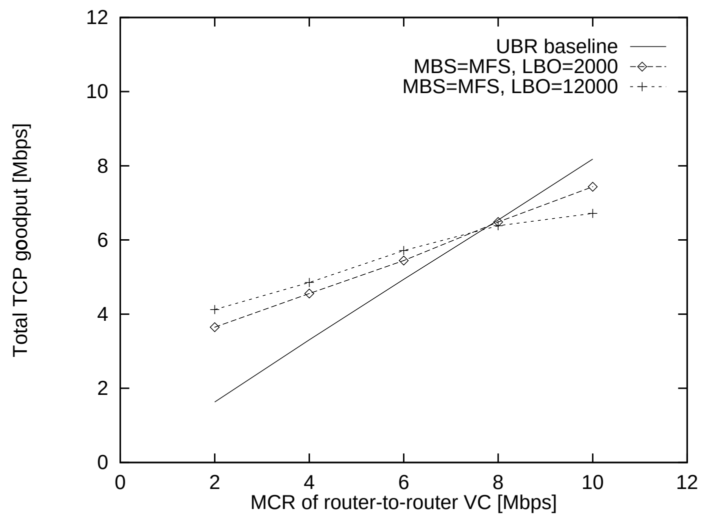

Figure 7.31: Five routers : Double EPD with LBO=2000 and LBO=12000

The influence of the LBO threshold on the total goodput of the sources attached to the different routers is mainly due to two factors. The first factor is that when the LBO threshold is set to 12000 cells, the packet loss ratio on the bottleneck switch is twice lower than with the LBO threshold set to 2000 cells. With a lower packet loss ratio, the TCP sources retransmit less packets and the total utilization of the backbone link increases. This factor explains the higher total goodput for the sources attached to a router with a small MCR. The second factor is that with a large LBO threshold, the traffic sent by the routers appears to be more bursty and less AAL5-PDUs are accepted by the F-GCRA at the ingress of the network, especially for the 10 Mbps router. There are slightly less CLP=0 AAL5-PDUs on the backbone link with a LBO threshold set to 12000 cells than with a LBO threshold set to 2000 cells. This means that more bandwidth is available for the CLP=1 AAL5-PDUs with a large LBO threshold. Furthermore, when we look at the CLP=1 AAL5-PDUs, it appears that the amount of CLP=1 AAL5-PDUs received by the destination routers is almost independent of their MCR. This means that the bandwidth which is not used by the CLP=0 AAL5-PDUs is shared equally among the five VCs.

Based on this simulation, one could recommend to set the LBO threshold to small values[7] since the performance decreases with an increase of the LBO threshold. However, it should be noted that setting the LBO threshold to a very small value would not be a good solution. With a LBO threshold set to 0 cell, all the non-eligible AAL5-PDUs would be discarded in the ATM switches. The simulations performed with this low threshold (figure 7.32) showed that the sources attached to each router had huge difficulties to utilize their reserved bandwidth, especially with a small MBS. These simulations confirm the recommendation found in the current definition of the GFR service category [Ken98], which notes that *"an implementation which discards all non-eligible AAL5-PDUs would be inconsistent with the "spirit" of the GFR service specification and would receive a poor service evaluation"*.

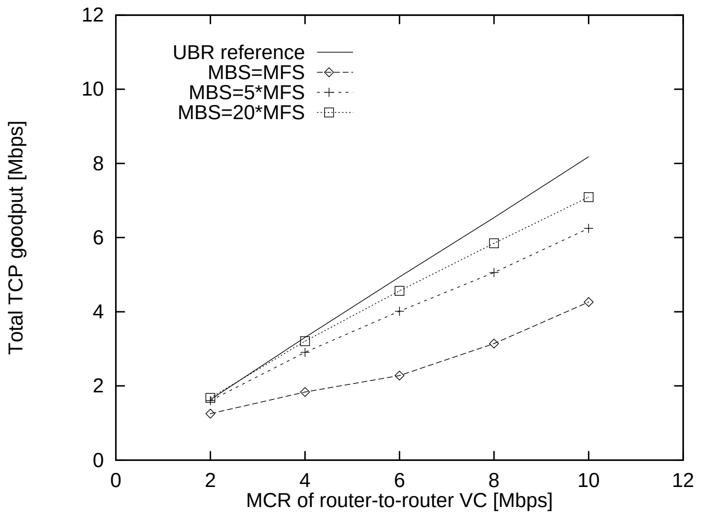

Figure 7.32: Five routers : Double EPD with LBO=0

From this simulation, it appears that the MBS of the GFR.2 traffic contracts should be set to a large value with the Double-EPD switch implementation.

Per-VC threshold and scheduling

For the simulations with the per-VC threshold and scheduling implementation, we assumed that a fraction of the buffer was reserved for each VC. For this, we varied the value of the T_i threshold and set the HBO threshold to $15000 - 5 \times T_i$ cells[8]. When the T_i threshold was set to small values (e.g. $MFS \leq T_i \leq 10 \times MFS$), our simulations did not reveal any difference in the total TCP goodput (figure 7.33). With a larger T_i threshold (e.g. $20 \times MFS$), the simulations showed a small decrease in the total goodput of the sources attached to the 10 Mbps router. A comparison with the UBR reference goodput shows that with the per-VC threshold and scheduling implementation the unreserved bandwidth on the backbone ATM link is distributed among the five VCs in proportion of their MCR.

[7]As already mentioned, this recommendation was proposed in [PB97] based on simulations performed with workstation traffic.

[8]The value of the LBO threshold is not relevant for the simulations discussed in this section since the routers only send CLP=0 AAL5-PDUs and no tagging is performed inside the network.

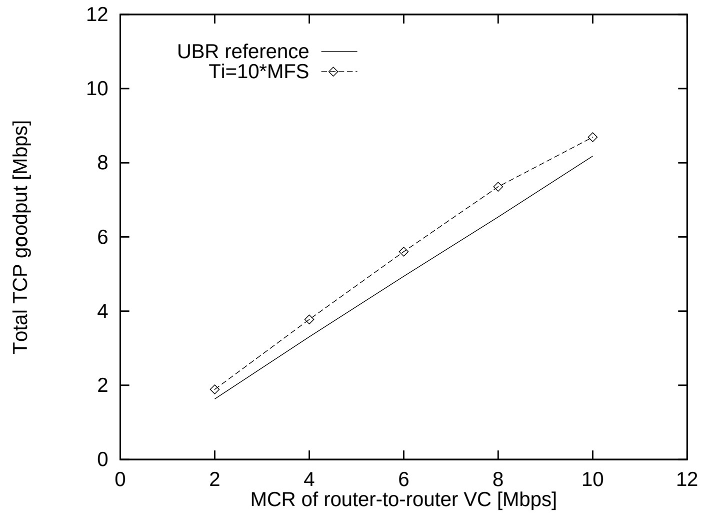

Figure 7.33: Five routers : per-VC threshold and scheduling

7.4.2 The ten routers scenario

As a reference for the "ten routers" scenario, we also consider a "reference" UBR scenario. In this UBR scenario, we artificially limit the bandwidth of the links between the second ATM switch and the R* routers to the MCR used for the GFR traffic contracts. The simulations performed with this modified scenario did not show any influence of the round-trip-time on the total goodput achieved by the TCP sources (figure 7.34). In all the figures of section 7.4.2 we show the total TCP goodput of the workstations attached to the various routers in function of the one-way delay on the router-to-router VC. In these figures, the curves with a dashed (resp. straight) line correspond to router-to-router VCs with a 4 Mbps (resp. 2 Mbps) MCR. We will use these UBR goodputs as a reference to compare the performance of the two GFR switch implementations.

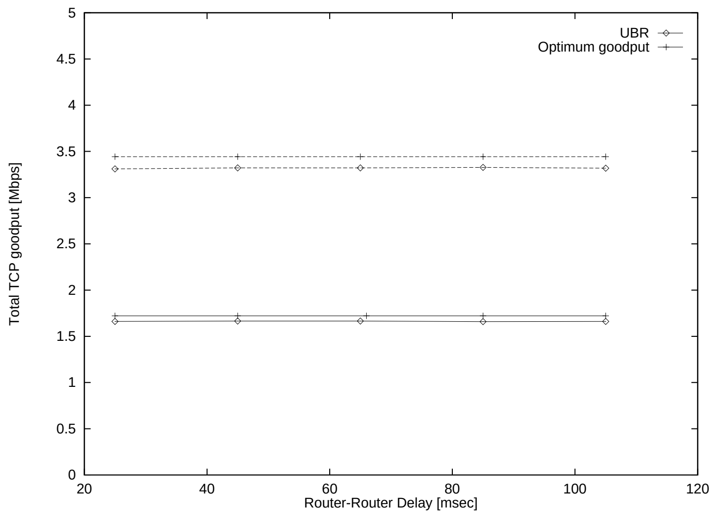

Figure 7.34: Ten routers : UBR reference

Double-EPD

For the simulations with Double EPD, we set the LBO threshold to 2000 cells and the HBO threshold to 15000 cells. As with the five routers scenario, the simulations with the ten routers scenario (figure 7.35) showed that the MBS of the GFR.2 traffic contract had a large influence on the total goodput achieved by the sources attached to each router.

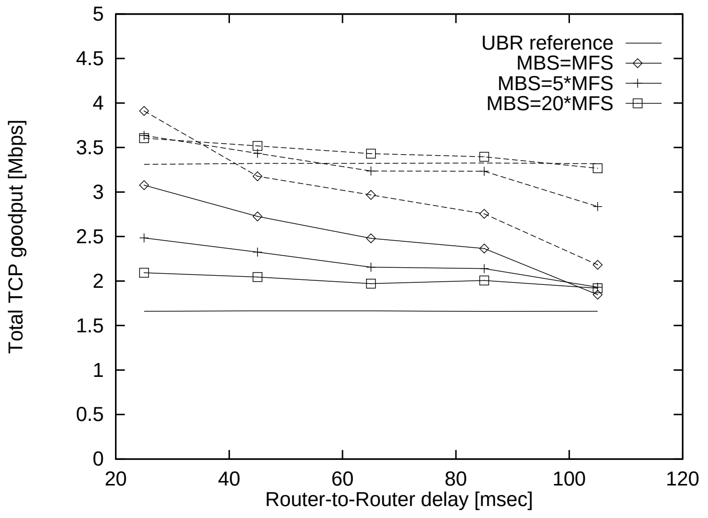

Figure 7.35: Ten routers : Double EPD

With a small MBS (MBS=MFS) for the GFR.2 traffic contract, the router-to-router delay has a large influence on the total goodput achieved by the sources connected to each router. With this small MBS, all the sources attached to a 2 Mbps router achieved a higher goodput than their UBR reference goodput, but this was at the prejudice of the sources attached to a 4 Mbps router with a large delay. With a larger MBS ($MBS = 20 \times MFS$), the sources attached to the 4 Mbps routers could utilize their reserved bandwidth and the influence of the router-to-router delay was limited. These simulations confirm the recommendation to utilize large values for the MBS with the Double-EPD switch implementation.

Per-VC threshold and scheduling

For the simulations with the per-VC threshold and scheduling implementation, we set the HBO threshold to $15000 - 10 \times T_i$ and varied the value of the T_i thresholds as with the five routers scenario. The simulations showed that the influence of the router-to-router delay with this switch implementation was not significant (figure 7.36). As with the five routers scenario, the unreserved bandwidth was distributed among the different VCs in proportion to their MCR.

7.4.3 The "mixed ten routers" scenario

As a reference for the "mixed ten routers" scenario, we consider the simulations performed with the per-VC threshold and scheduling implementation with the "five routers" scenario.

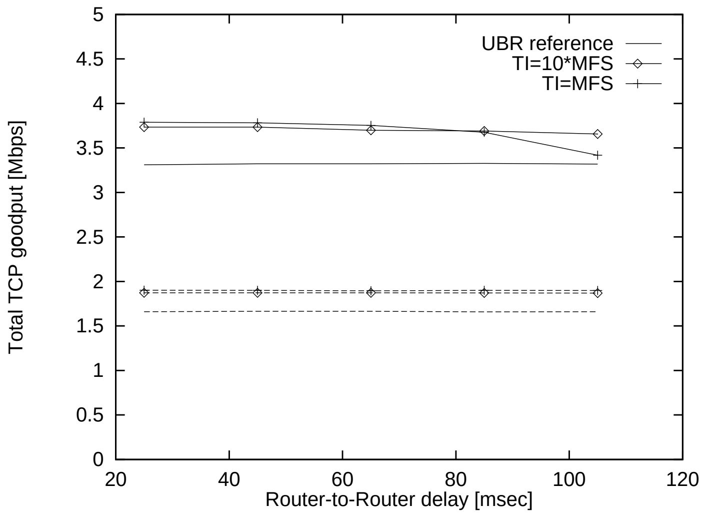

Figure 7.36: Ten routers : per-VC threshold and scheduling

If the GFR conformance definition did not have an influence on the total TCP goodput, then the various VCs should achieve the same goodput as in the "five routers" scenario. We consider the total goodput of the "five routers" scenario (see figure 7.33) as a reference goodput for this section.

For the simulations with the "mixed ten routers" scenario, we used the same MBS for the traffic contracts of the GFR.1 and GFR.2 VCs and set the T_i thresholds equal to the MBS. We then set the LBO threshold to 6000 cells and the HBO threshold to $15000 - 10 \times T_i$ cells. The simulations performed with the MBS of the traffic contracts equal to the MFS showed a huge unfairness (figure 7.37). With this small MBS, the GFR.2 VCs cannot utilize their minimum guaranteed bandwidth and consequently the GFR.1 VCs receive a large fraction of the available bandwidth.

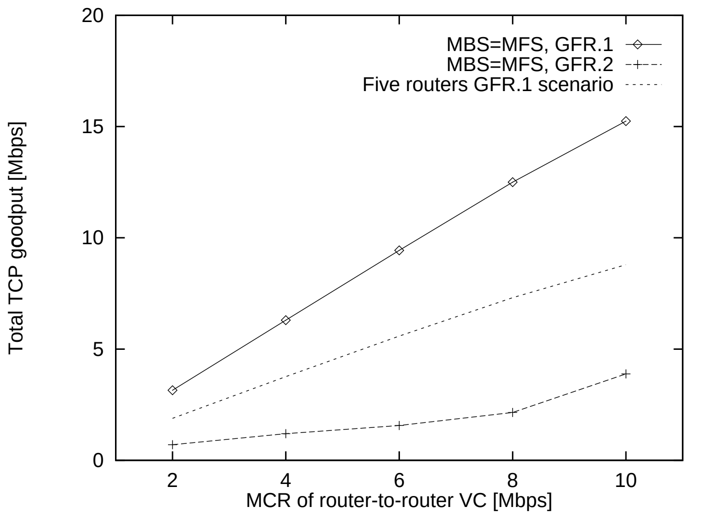

Figure 7.37: Mixed ten routers scenario with MBS=MFS

The unfairness of the per-VC threshold and scheduling implementation with the "mixed ten routers" scenario is due to several factors. The first factor, as already mentioned previously for the simulations with the other scenarios and the Double-EPD implementation, is that the F-GCRA used at the ingress of the network with the GFR.2 conformance definition tags a large fraction of the AAL5-PDUs. For example, with the MBS equal to the MFS (32 cells), 28% of the AAL5-PDUs sent by the 2 Mbps GFR.2 router and 11% of the AAL5-PDUs sent by the 10 Mbps GFR.2 router are tagged. The second factor is the buffer acceptance algorithm used by the per-VC threshold and scheduling implementation. This buffer acceptance algorithm discards the arriving CLP=1 AAL5-PDUs when the buffer occupancy is above the LBO threshold, but the arriving CLP=0 AAL5-PDUs are only discarded when the buffer occupancy is above the HBO threshold and the per-VC occupancy is above the T_i threshold. During our simulations, none of the CLP=0 AAL5-PDUs sent by the GFR.2 routers was discarded at the bottleneck switch. However, almost all the AAL5-PDUs tagged by the F-GCRA on the GFR.2 VCs were discarded on the bottleneck switch. This is due to the fact that the buffer of the bottleneck switch was entirely filled with CLP=0 AAL5-PDUs sent by the GFR.1 VCs. In fact, the average total buffer occupancy of the bottleneck switch was close to 13000 cells while the average occupancy of the per-VC queues of the GFR.2 VCs was very small. These combined two factors are the main reason for the low performance of the per-VC threshold and scheduling implementation in this case.

It should be noted that with the "mixed ten routers" scenario and a small MBS, the GFR.2 routers could not utilize their minimum guaranteed bandwidth. For example, the total goodput of the sources attached to the 10 Mbps GFR.2 router was only 3.9 Mbps, while the minimum guaranteed bandwidth when taking into account the TCP/IP/AAL5 and ATM overheads is 8.6 Mbps. This is a very low utilization of the minimum guaranteed bandwidth.

The simulations with a MBS set to $20 \times MFS$ (figure 7.38) showed a similar but less severe unfairness. In this case, the fraction of the AAL5-PDUs tagged by the F-GCRA on the GFR.2 was smaller than with the smaller MBS. For example, 20% of the AAL5-PDUs sent by the 2 Mbps GFR.2 router and only 9% of the AAL5-PDUs sent by the 10 Mbps GFR.2 router were tagged by the F-GCRA. In this case also, almost all the CLP=1 AAL5-PDUs were discarded on the bottleneck switch.

With a larger MBS, the sources attached to the GFR.2 routers could not still efficiently utilize their reserved bandwidth. For example, the total goodput for the sources attached to the 10 Mbps GFR.2 router was only 6.6 Mbps instead of 8.6. While the total goodput of the sources attached to the GFR.2 routers increases, using a large MBS is probably not a solution to efficiently support both GFR.1 and GFR.2 VCs with the per-VC threshold and scheduling implementation.

In fact, when used with both GFR.1 and GFR.2 VCs, the per-VC threshold and scheduling implementation behaves almost like the Double-EPD implementation with a LBO threshold set to 0 cells since in both cases all the AAL5-PDUs tagged by the F-GCRA are discarded. This is annoying since the per-VC threshold and scheduling implementation is much more complex than the Double-EPD implementation.

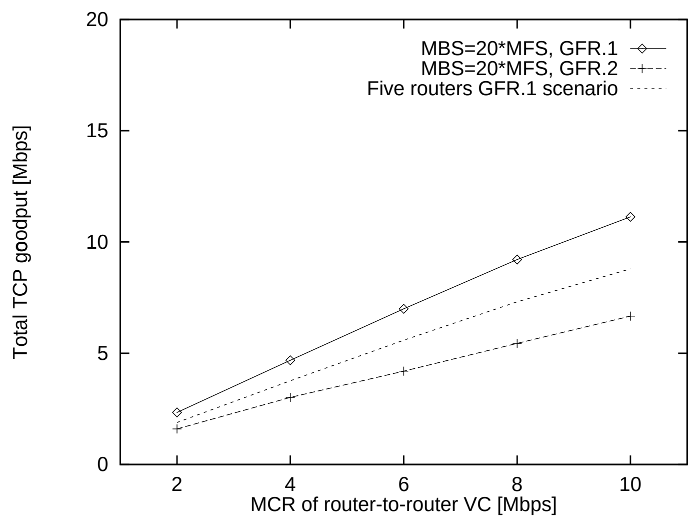

Figure 7.38: Mixed ten routers scenario with MBS=20 $\times$ MFS

7.4.4 Related work

Most of the simulations study on the performance of TCP with the GFR service category have considered workstation traffic (see section 7.3.7). However, recently several papers and ATM Forum contributions [HV98a, Lee98, GJFV98] have, like us [Bon98b], also considered internetwork traffic.

In [HV98a], the authors compare the performance of the Double-EPD implementation for GFR.2 with Partial Buffer Sharing for VBR.3. The simulations discussed in [HV98a] consider a single simulation scenario similar to our "five routers" scenario. They study the influence of the MBS of the GFR traffic contracts on the performance of TCP. The simulations show that GFR.2 provides a better performance than VBR.3 and that a large MBS is required to allow the sources attached to a router with a large MCR to efficiently utilize the minimum guaranteed bandwidth. These conclusions are in line with our simulations with the "five routers" scenario. Compared with the simulations discussed in [HV98a], our simulations have also studied the performance of the per-VC threshold and scheduling implementation and considered other network scenarios than the "five routers" scenario.

[GJFV98] proposes a new buffer acceptance mechanism to support the GFR service category in FIFO-based ATM switches with per-VC accounting and studies its performance with internetwork traffic. We discuss this proposed implementation in section 8.3.6.

7.5 Drawbacks of the proposed implementations

The simulations discussed in the previous sections have allowed us to gain a good understanding of the behaviour of the two example GFR switch implementations [Ken98] with TCP/IP traffic. Based on the results of these simulations, we identify in this section the main drawbacks of the Double-EPD and of the per-VC threshold and scheduling switch implementations.

7.5.1 Drawbacks of Double-EPD

The Double-EPD implementation suffers from three important drawbacks. The first one is that it was designed with the GFR.2 conformance definition in mind. The second drawback is that it provides a relatively low performance when used with workstation and to a smaller extent internetwork traffic. The third drawback is that it cannot isolate one VC from the behaviour of other VCs.

Support for GFR.1

The Double-EPD implementation requires an explicit identification of the eligible and the non-eligible AAL5-PDUs. With the GFR.2 conformance definition, this identification is performed by the F-GCRA at the ingress of the network. It can be expected that the Double-EPD implementation will be mainly used to support the GFR.2 conformance definition. However, it might be possible to combine a F-GCRA with a Double-EPD implementation to support the GFR.1 conformance definition. In some ATM switches, the internal representation of the ATM cell is slightly larger than the standard 53 bytes. This internal representation usually carries internal parameters for the switch (e.g. a routing tag for self-routing switches). If there is at least one unused bit in this internal representation, then it is possible to support the GFR.1 conformance definition by combining a F-GCRA with a trivial modification to the Double-EPD implementation. The F-GCRA would be used at the ingress of this switch to set the unused bit of the internal representation to 1 (resp. 0) for the non-eligible (resp. eligible) AAL5-PDUs. The Double-EPD implementation would then decide whether to accept an arriving AAL5-PDU on the basis of the occupancy of its buffer and the value of the “unused” bit for the first cell of the AAL5-PDU. These internal representations of the ATM cells are common in large switches (e.g. when a routing tag is attached to each cell), but finding a spare bit in this internal representation is not necessarily always possible in those switches.

Performance with TCP/IP traffic

The simulations discussed in sections 7.3 and 7.4 have shown that the Double-EPD implementation, combined with the F-GCRA, has difficulties to efficiently support TCP/IP traffic. This low efficiency is mainly due to the burstiness of TCP/IP traffic which causes the F-GCRA to tag a large fraction of the AAL5-PDUs. We will however show in section 8.2 that the performance of Double-EPD when carrying workstation and internetwork traffic can be significantly improved with a relatively simple modification to the ATM adapters.

Lack of VC isolation

The third drawback of the Double-EPD implementation is the lack of VC isolation for the CLP=1 traffic. This lack of isolation is not a problem if all the sources are cooperative and adapt their transmission rate when congestion occurs. This is the case we studied with workstation and internetwork traffic in our simulations. However, a non-cooperative source could utilize a large fraction of the unreserved bandwidth by forcing the other cooperative sources to unnecessarily reduce their transmission rates. The Double-EPD implementation does not provide any mechanism to control such non-cooperative sources. This is not desirable, especially in public networks where it cannot be assumed that all the sources will cooperate.

Another problem related to this lack of isolation is that the network operator has no control on how the unreserved bandwidth is distributed among the active VCs.

7.5.2 Drawbacks of per-VC threshold and scheduling

The simulations discussed in the previous sections have shown that the per-VC threshold and scheduling implementation had several difficulties to efficiently support both GFR.1 and GFR.2 VCs at the same time. This problem is in fact related to the interpretation of the relative importance of CLP=1 AAL5-PDUs compared with non-eligible CLP=0 AAL5-PDUs. This interpretation is currently not addressed in [Ken98, SG198], but is important when considering the deployment of GFR. The second drawback, which did not appear in the simulations, is the inefficient support of GFR VCs with MCR=0 by the current WFQ-like schedulers due to the fact that it is impossible to use a weight of 0 for these VCs.

Importance of the CLP=1 AAL5-PDUs

The question of the importance of the CLP=1 AAL5-PDUs will become clear by considering a very simple example. Let us assume a single congested switch based on the per-VC threshold and scheduling implementation. Let us also assume that two GFR.1 VCs (VC_A and VC_B) are established through this switch and that both VCs have a MCR equal to 50% of the output link of the switch. The only difference between the two VCs is that VC_A is continuously sending CLP=0 AAL5-PDUs while VC_B is continuously sending CLP=1 AAL5-PDUs. In this case, 50% of the AAL5-PDUs from VC_A are eligible for the minimum guaranteed bandwidth and thus VC_A should achieve a throughput which is at least equal to its MCR. For VC_B, its CLP=1 AAL5-PDUs are not eligible for the minimum guaranteed bandwidth. Thus, according to the definition of the GFR service category, 50% of the output link must be used to carry 50% of the (CLP=0) AAL5-PDUs from VC_A. However, the definition of the GFR service category does not specify how the remaining 50% of the output link should be shared between the non-eligible (CLP=0) AAL5-PDUs from VC_A and the non-eligible (CLP=1) AAL5-PDUs from VC_B. There are two possible interpretations of the importance of CLP=1 AAL5-PDUs in a real ATM switch.

The first interpretation, which we will call strict CLP priority, is to consider that a CLP=1 AAL5-PDU is always less important[9] than a CLP=0 AAL5-PDU, even if this AAL5-PDU is non-eligible for the minimum guaranteed bandwidth and belongs to a different VC. The per-VC threshold and scheduling implementation supports this interpretation.

This strict CLP priority is probably mainly suitable for private networks where the network manager has a complete control on the devices attached to the ATM network. In such a private network, the network manager may wish to utilize the CLP=0 AAL5-PDUs

[9]By "less important", we mean that if a switch has to discard one AAL5-PDU due to internal congestion, then it should preferably discard the less important one.

only for mission-critical traffic and thus require that CLP=0 AAL5-PDUs should only be discarded when congestion is exclusively caused by CLP=0 AAL5-PDUs. If the network manager chooses this approach, it can however be expected that the total amount of CLP=0 AAL5-PDUs in the network would still be limited by some means. We list below some possible applications for the strict CLP priority :

- In a corporate Intranet, some traffic may be much more valuable (at application level) than other traffics. A typical example is SNA traffic from mainframes. Other examples are mirroring of mainframes or transaction processing. Usually the amount of traffic generated by these applications is limited compared with the bandwidth available on the Intranet.

- In an ISP network, the traffic which is received from the expensive wide area links (e.g. transatlantic links) has a higher monetary cost for the ISP operator than the local traffic. It would thus be desirable to protect the AAL5-PDUs which have crossed the expensive wide area links over the AAL5-PDUs generated locally. This can be done by using only CLP=0 AAL5-PDUs for the packets received from the wide area links and using CLP=1 AAL5-PDUs for the traffic which does not cross the border of the ISP network. Since the bandwidth on the wide area links is usually much lower than the bandwidth available on the ISP network, then the total amount of CLP=0 AAL5-PDUs is limited.

In these examples, the total amount of CLP=0 AAL5-PDUs is limited by an explicit configuration of all the devices attached to the ATM network by the network manager. Thus, while short term congestion may cause CLP=0 AAL5-PDUs to starve traffic from CLP=1 AAL5-PDUs, it can be expected that on average a large amount of the available bandwidth would be left for the CLP=1 AAL5-PDUs.

The second interpretation, which we will call relative CLP priority, is to consider that a CLP=1 AAL5-PDU is always less important than a CLP=0 AAL5-PDU from the same VC, but that a CLP=1 AAL5-PDU from one VC (e.g. a VC with a high MCR) might be more important than a CLP=0 AAL5-PDU from another VC (e.g. a VC with MCR=0). This second interpretation is suitable for public networks where the charging is likely to be only based on the traffic contract (e.g. MCR) and the duration of the VC, but does not depend on the actual volume of CLP=0 and CLP=1 cells. In this case, a user with a large MCR should always be able to successfully transmit data at the negotiated MCR, even during short periods of time where he is only sending CLP=1 AAL5-PDUs.

The relative CLP priority is probably preferable[10] in public networks which supports the two GFR conformance definitions. In this case, favoring the GFR.1 VCs over the GFR.2 VCs like the per-VC threshold and scheduling implementation (see section 7.4.3) is probably not an acceptable solution. The relative CLP priority is not supported by the per-VC threshold and scheduling implementation.

[10]While we were proofreading this thesis, [WNH98] proposed to clarify the GFR service category definition by adopting the relative CLP priority interpretation.

Support for best-effort VCs

The simulations discussed in the previous sections have shown that the Virtual Spacing scheduler distributes the available bandwidth to all the VCs in proportion to their respective MCR. In fact, most of the WFQ-like schedulers discussed in the literature distribute the available bandwidth in a similar way. Such a distribution of the available bandwidth among the active VCs might be suitable in some networks, but it is not appropriate for all environments. Some network operators would probably want to distribute the unreserved bandwidth differently than in proportion to the MCR of the active VCs.

Furthermore, to support best-effort (i.e. UBR or GFR with MCR=0) VCs in switches using such schedulers it is necessary to implicitly reserve some bandwidth for these best-effort VCs. If each best-effort VC is mapped into a single queue which is served by a WFQ-like scheduler, then a non-zero MCR must be associated with each queue and thus with each best-effort VC. If all the best-effort VCs are mapped into a single queue which is served by a WFQ-like scheduler, then a non-zero MCR must be associated with the group of best-effort VCs. In both cases, the MCR which is implicitly associated with the best-effort VCs is not available to carry traffic from the GFR VCs with a non-zero MCR.

In order to reduce the impact of this implicit bandwidth reservation, one could think of setting the MCR associated with the group of best-effort VCs to the lowest value supported by the WFQ-like scheduler. Unfortunately, this solution is not without problems. For example, let us assume that the smallest MCR supported by the WFQ-like scheduler is equal to one cell per second. In this case, the bandwidth which is implicitly reserved for the best-effort VCs is very small and thus it does not severely limit the amount of bandwidth which can be reserved for the GFR VCs with a non-zero MCR. Let us also assume that the WFQ-like scheduler is attached to a 34 Mbps (80000 cells/sec) E-3 ATM link and that one GFR VC with a MCR of 1000 cells per second is using this link together with one best-effort VC. If both VCs are always continuously active, then the WFQ-like scheduler will serve the best-effort VC at a rate of $80000 \times \frac{1}{1001} = 80$ cells per second while the VC with a 1000 cells per second MCR will be served at a rate of 79920 cells per second. This cannot be considered as an acceptable support for the best-effort VCs.

On the other hand, if the switch had implicitly assigned all the unreserved bandwidth to the best-effort VCs, then no unreserved bandwidth would be available for the VCs with a minimum guaranteed bandwidth. This solution is not more acceptable than the previous one.

The discussion above has shown that distributing the unreserved bandwidth in proportion to the MCR of the active VCs like current WFQ-like schedulers causes several problems when VCs with and without a minimum guaranteed bandwidth are multiplexed on a single link. A more flexible distribution of the unreserved bandwidth is desirable to efficiently support GFR VCs in an ATM switch.

7.6 Conclusion

In this chapter, we have presented two important contributions concerning the utilization of the GFR service category to provide services with a minimum guaranteed bandwidth to TCP/IP traffic.

Our first contribution is a detailed simulation study of the performance of the Double-EPD and the per-VC threshold and scheduling implementations with workstation traffic [Bon98d]. For this simulation study, we have considered three TCP implementations with different retransmission mechanisms. Our simulations in LAN and WAN environments have shown that TCP had huge difficulties to utilize the minimum guaranteed bandwidth of the ATM VC with the Double-EPD implementation. The performance of the per-VC threshold and scheduling implementation was much better, but there were still some performance problems, especially in WAN environments and when the GFR.2 conformance definition was used. This simulation study [Bon97] was one of the first detailed simulation study of workstation traffic with the GFR service category and was largely discussed within the ATM Forum and the ITU-T when it was publicly distributed in July 1997.

Our second contribution is a detailed simulation study of the performance of the two switch implementations mentioned above with internetwork traffic [Bon98b]. This simulation study has shown that the two switch implementations provide a better performance with internetwork traffic than with workstation traffic. However, the simulations with the Double-EPD implementation have shown that large values for the MBS were required to efficiently support internetwork traffic. The performance of the per-VC threshold and scheduling implementation was always good with GFR.1 VCs. However, our simulations have shown that the per-VC threshold and scheduling implementation had huge difficulties to efficiently support both GFR.1 and GFR.2 VCs in the same switch. In this case, our simulations have shown that the GFR.2 VCs were not able to utilize their minimum guaranteed bandwidth while the GFR.1 VCs could grab some of the bandwidth reserved for the GFR.2 VCs.

Based on the lessons learned from the simulations with workstation and internetwork traffic, we have then identified several drawbacks of the Double-EPD and per-VC threshold and scheduling implementations. We have explained the two possible interpretations of the CLP bit with the current definition of the GFR service category. We have also shown that distributing the unreserved bandwidth in proportion to the MCR of the active VCs as done by most WFQ-like schedulers could lead to difficulties to support GFR VCs with a minimum guaranteed bandwidth together with best-effort VCs.

We will investigate in the following chapter several possible solutions to improve the performance of the GFR switch implementations with TCP/IP traffic.

Chapter 8

Improvements to the GFR switch implementations

8.1 Introduction

The simulations described in the previous chapter have allowed us to gain a much better understanding of the behaviour of the Double-EPD and per-VC threshold and scheduling implementations when carrying TCP/IP traffic. Based on this knowledge, we investigate in this chapter three possible ways to improve the performance of GFR switch implementations when carrying TCP/IP traffic.

This chapter is structured as follows. We first evaluate in section 8.2 whether the performance of TCP/IP with the GFR.2 conformance definition can be improved by adding some traffic shaping capabilities to the ATM adapters. We then investigate in section 8.3 whether it is possible to completely support the GFR service category in FIFO-based ATM switches without requiring per-VC queueing and scheduling. Finally we propose in section 8.4 an extension to the Virtual Spacing algorithm to better support the GFR service category with the per-VC threshold and scheduling implementation.

8.2 Improving the performance of TCP with GFR.2

The simulations discussed in the previous chapter have shown that the main drawback of the GFR.2 conformance definition is that, due to the burstiness of workstation and internetwork traffic, the F-GCRA at the ingress of the network may tag a large fraction of the AAL5-PDUs even if the long-term average rate of the GFR VC is much lower than its minimum guaranteed bandwidth. This large number of CLP=1 AAL5-PDUs causes performance problems when the GFR.2 conformance definition is used with the Double-EPD or the per-VC threshold and scheduling implementations.

In this section, we investigate whether it is possible to improve the performance of workstation and internetwork traffic with the GFR.2 conformance definition by reducing the burstiness of the traffic with a simple shaping algorithm. We present a simple shaper suitable for the GFR service category in section 8.2.1 and discuss its performance with workstation and internetwork traffic in sections 8.2.2 and 8.2.3.

8.2.1 A simple shaper

Shaping algorithms have already been proposed several times to improve the performance of ATM networks. One of the first utilizations of shaping algorithms was to reduce the Cell Delay Variation (CDV) on CBR VCs [BGSC92]. It was shown that by reducing this CDV it was possible to increase the utilization of the ATM links while preserving the QoS guarantees. Several implementations of such shapers based on the GCRA algorithm were proposed [WW92, BSG92]. Similar shapers were also proposed to support the VBR service category [LAH97].

Another popular application of shaping algorithms is to reduce the burstiness of video traffic. In this case, the objective of the shaper is to reduce the burstiness of the traffic while maintaining the QoS guarantees and notably the delay guarantees. Several types of shapers have been proposed in the literature. Some control the transmission rate by taking into account a model of the video traffic (see e.g. [CT98]). Others control both the transmission rate and some parameters of the video compression scheme with a feedback loop between the shaper and the video coder (see e.g. [HRR97]).

In these applications, the shaping algorithm is mainly used to ensure that the traffic at the output of the shaper is conformant with a predefined CBR or VBR traffic contract. This means that the burstiness of the traffic is bounded by the parameters of the traffic contract. When considering a shaper to support the GFR service category, we would like to reduce the burstiness of the traffic while still allowing the GFR VCs to benefit from any available unreserved bandwidth inside the network. For this, we consider a stand-alone shaper as shown in figure 8.1. This shaper contains a buffer and is able to regulate its transmission rate. In practice we expect that the shaper would be integrated inside the ATM adapters of the workstations or the routers or used up-front of the F-GCRA in a GFR policing unit.

Since the goal of the simple shaper is to reduce the burstiness of the traffic, a simple solution to regulate the transmission rate of the shaper would be to measure the long-term average arrival rate of the incoming traffic and to set the transmission rate equal to this average rate. Although this solution would clearly limit the burstiness of the traffic at the output of the shaper, it would also probably result in a very large occupancy of the buffer of the shaper. Such a large occupancy would cause large queueing delays and could not be considered as an acceptable solution. A real shaper should be able to reduce the burstiness of the traffic with a small buffer and of course without cell losses.

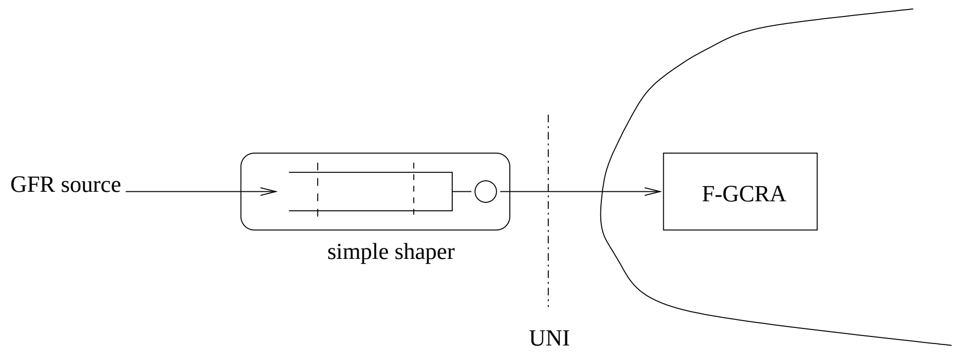

Figure 8.1: Reference configuration for the simple shaper

For a simple shaper, we propose to compute the transmission rate of the shaper as shown in equation 8.2.1, where B_{Occ} is the occupancy of the shaper buffer and ACR is the long term measured average incoming rate of the traffic. We assume that the endsytems are only sending CLP=0 AAL5-PDUs and consider two thresholds on the occupancy of the shaper buffer : Min_{th} and Max_{th}. When the occupancy of the shaper buffer is below Min_{th}, the transmission rate of the shaper is set to*max(ACR,MCR)*. We use the maximum of the long term average rate and the minimum guaranteed bandwidth to allow the shaper to always transmit at least at the minimum guaranteed bandwidth. This is especially important when an endsystem starts to transmit since at that time ACR is not yet a valid indication of the long term average rate. We expect that usually the long term average rate will be larger than the MCR. When the occupancy of the shaper buffer is larger than or equal to Max_{th}, the buffer is heavily filled and to avoid cell losses, we set the transmission rate to the PCR of the GFR traffic contract. If the incoming traffic conforms to the PCR part of the GFR traffic contract, then the occupancy of the shaper buffer should not exceed Max_{th}. Thus, Max_{th} would usually be close[1] When the occupancy of the shaper buffer is between Min_{th} and Max_{th}, the transmission rate is set to the maximum of the measured average rate and a linear function of the buffer occupancy. The transmission rate of the shaper is also shown graphically in figure 8.2. The transmission rate is computed every time a cell is transmitted by the shaper.

$$\begin{cases} B_{Occ} \leq Min_{th} & \Rightarrow max(ACR, MCR) \\ Min_{th} < B_{Occ} \leq Max_{th} & \Rightarrow max(ACR, MCR + \frac{PCR-MCR}{Max_{th}-Min_{th}} \times (B_{Occ} - Min_{th})) \\ Max_{th} < B_{Occ} & \Rightarrow PCR \end{cases} \tag{8.1}$$

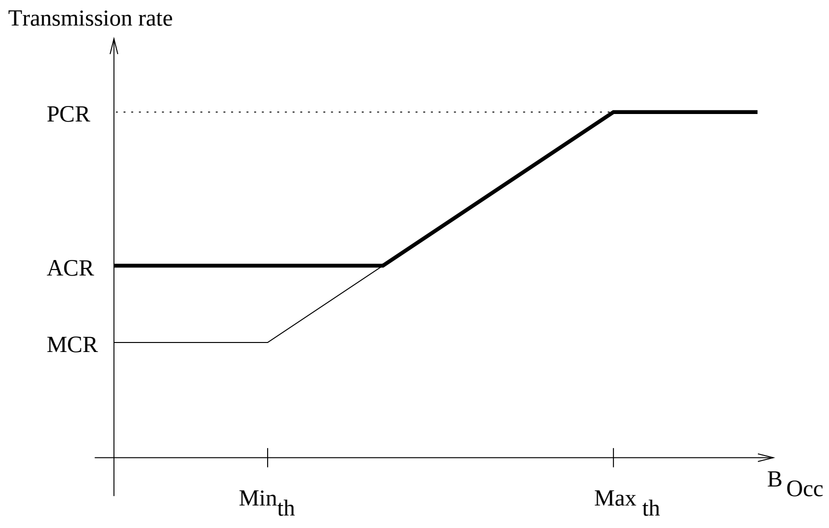

Figure 8.2: Simple function to compute the transmission rate of the shaper

[1]The difference between Max_{th} and the size of the buffer would depend on the amount of CDV tolerance associated with the PCR of the traffic contract.

As a simple solution to compute the average arrival rate of cells in the shaper buffer, we measure the number of cells which arrive during a period of T seconds and compute the average rate as $ACR = ACR - wq \times (ACR - cells/T)$ where$cells$ is the number of cells received during the last period of T seconds and wq an averaging factor. ACR is set equal to the MCR of the GFR traffic contract when the shaper is initialized.

8.2.2 Performance with workstation traffic

To evaluate whether a simple shaper such as the one discussed above could improve the performance of workstation traffic with the Double-EPD implementation, we considered the LAN environment studied in section 7.3. We modified the simulation scenario by inserting a simple GFR shaper with 2000 cells of buffer at the output of each workstation. The Min_{th} threshold of the shaper was set to one maximum sized frame (192 cells) and Max_{th} was set to 2000 cells. We choose a non-zero value for Min_{th} in order to ensure that the simple shaper can absorb a small burst of cells while still transmitting at the long term average rate. The window size for the ACR measurements was set to 10 milliseconds and the averaging factor (wq) was set to 0.05. In practice, we expect that the shaping would be performed by the ATM adapter under the control of the ATM device driver. Besides the introduction of the simple shaper, we did not modify the simulation scenario used in section 7.3. We considered one privileged source and varied its MCR from 20000 cells per second to 180000 cells per second and nine background sources with their MCR set to 20000 cells per second. We only consider TCP_{SACK} for the simulations with the simple shaper since the simulations performed without a shaper showed that this implementation achieved the best performance.

The simulations performed with the simple shaper (figure 8.3) revealed a significant performance gain due to the introduction of the shaper. With the simple shaper, the privileged source was almost always able to completely utilize its guaranteed minimum bandwidth. The benefits of the simpler shaper are particularly visible when we compare the results shown in figure 8.3 with those shown previously in figure 7.10 (see page 107). Without a shaper, the throughput of the privileged source did not reach 30 Mbps with a MCR of 180000 cells per second while it reached 65 Mbps with a shaper.

We also performed simulations with larger values for Min_{th}. These simulations showed that the throughput of the privileged source decreased as Min_{th} increased, especially with a large MSS. With a MSS set to 9140 bytes and a larger Min_{th}, the F-GCRA tagged more AAL5-PDUs than with a smaller Min_{th}. Some of these tagged AAL5-PDUs were discarded by the Double-EPD switch and these packet drops resulted in expirations of TCP's retransmission timer. With a MCR set to 180000 cells per second, the retransmission timer of the privileged source expired on average every five seconds with Min_{th} set to 192 cells and every 2 seconds with Min_{th} set to 1000 cells. In both cases, Max_{th} was set to 2000 cells. With a MSS of 1460 bytes there were few expirations of the retransmission timer, even with larger values for Min_{th}. This is not surprising since the simulations discussed in section 7.3 had already shown that TCP_{SACK} suffered from less expirations of its retransmission timer with a 1460 bytes MSS than with a 9140 bytes MSS.

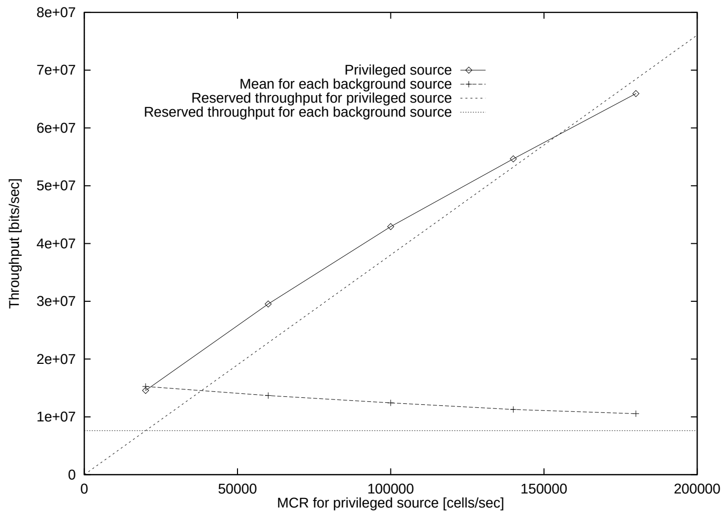

Figure 8.3: TCP_{SACK} throughput for privileged and background sources with simple shaper

8.2.3 Performance with internetwork traffic

To evaluate the suitability of the simple shaper to efficiently support internetwork traffic, we considered the five routers scenario with the Double EPD switch implementation. The only modification we introduced in the simulation scenario was that we attached a simple shaper to the output queue of each router. The shaper was configured with a Max_{th} threshold of 2000 cells and we varied the value of the Min_{th} threshold. To reduce the number of variable parameters, we set the Min_{th} threshold equal to the MBS of the traffic contract of the GFR VCs. We used a LBO threshold of 2000 cells with the Double-EPD implementation. The simulations performed with the shaper (figure 8.4) showed that the workstations attached to the various routers were able to efficiently utilize their MCR. This is a significant improvement over the simulations performed without the simple shaper (see figure 7.30 on page 126). Furthermore, the simulations showed that with the simple shaper, the goodput achieved by the workstations attached to the routers was almost independent of the MBS of their traffic contracts. This means that with the simple shaper it is possible to achieve good performance with the Double-EPD implementation and small MBS values. A closer look at the simulations showed that with the simple shaper, the unreserved bandwidth was almost equally shared among all the active VCs.

We also performed simulations with the LBO threshold of the Double-EPD switches set to 0. In this case, all the CLP=1 AAL5-PDUs are discarded by the ATM switches and simulations performed without a shaper (see figure 7.32 on page 128) have shown that the workstations attached to the various routers had huge difficulties to utilize their minimum guaranteed bandwidth, especially when a low value for the MBS was used. With a simple shaper attached to each router, the simulations (figure 8.5) showed that the workstations were able to utilize their minimum guaranteed bandwidth. Again, with the simple shaper, the total goodput is almost independent of the MBS of the GFR traffic contracts. This is a significant improvement compared to the simulations without the simple shaper.

Finally, we considered the "mixed ten routers" scenario. In this scenario, we compare

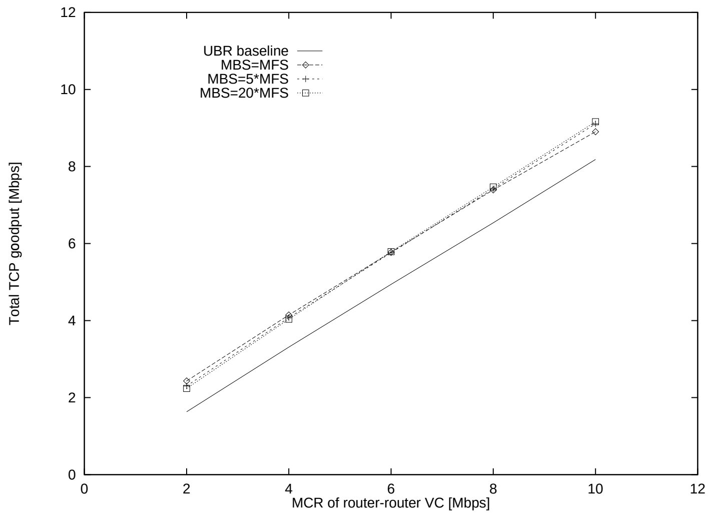

Figure 8.4: Five routers scenario with simple shaper and LBO=2000

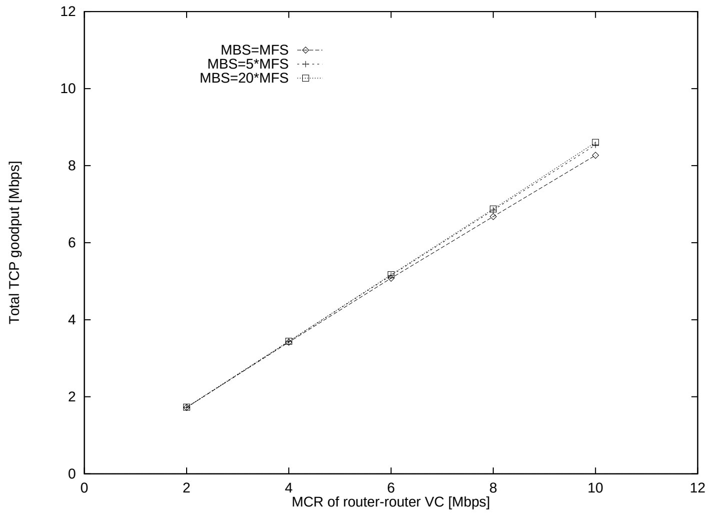

Figure 8.5: Five routers scenario with simple shaper and LBO=0

the performance of the per-VC threshold and scheduling implementation with GFR.1 and GFR.2 VCs. In this case, the simulations performed without a simple shaper (see figure 7.37 on page 131) have shown a huge unfairness between the GFR.1 and the GFR.2 VCs, especially with a small MBS. The simulations with a simpler shaper attached to each GFR.2 router and a MBS of 32 cells (figure 8.6) showed that the simple shaper allowed the GFR.2 routers to correctly utilize their minimum guaranteed bandwidth. The simulations with a larger MBS produced similar results. The main reason for the improved performance was that with the shaper the GFR.2 routers were able to send CLP=0 AAL5-PDUs at a rate which is close to their minimum guaranteed bandwidth. Of course, the GFR.2 routers were still unable to benefit from the unreserved bandwidth since the per-VC threshold and scheduling implementation supports a strict interpretation for the CLP bit and discards all the CLP=1 AAL5-PDUs from the GFR.2 VCs. A modification to the buffer acceptance mechanism used by the per-VC threshold and scheduling implementation would be necessary to avoid this unfairness by supporting a relative interpretation for the CLP bit instead of a strict interpretation.

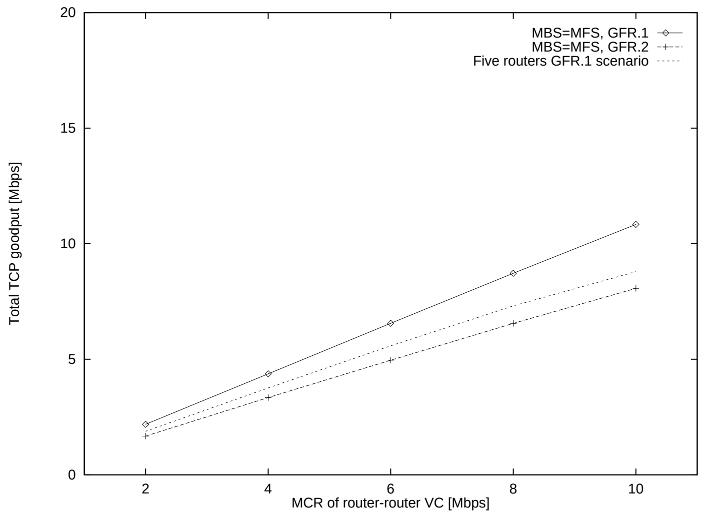

Figure 8.6: Mixed ten routers scenario with simple shaper and MBS=32

8.2.4 Further work

The simulations discussed above have clearly shown that shaping could be an interesting solution to improve the performance of TCP/IP traffic with the GFR.2 conformance definition. If the GFR.2 conformance definition becomes widely used, it might be worthwhile to try to improve the performance of the simple shaper discussed in this section. A first way to improve the shaper is the computation of the average rate. We used a fixed window in the simple shaper, but this is probably not the best solution. Possible averaging functions include the exponential averaging proposed by [SSZ98] in another context or moving windows. A second way to improve the simple shaper would be to explore other solutions than the linear function used by the simple shaper to compute the transmission rate as a function of the occupancy of its buffer. Finally, the shaper should treat differently the CLP=1 and the CLP=0 AAL5-PDUs. A possible way to do this would be to always transmit the CLP=1 AAL5-PDUs at PCR and to use the shaper for the CLP=0 AAL5-PDUs only[2] . By sending the CLP=1 AAL5-PDUs at PCR we limit the interactions between the CLP=0 and CLP=1 AAL5-PDUs in the buffer of the shaper. Since the goal of the shaper is mainly to reduce the burstiness of the CLP=0 cell flow in order to limit the number of cells which are tagged by the F-GCRA, sending CLP=1 AAL5-PDUs at PCR allows the shapper to better regulate the flow of CLP=0 AAL5-PDUs. Another way to improve the simple shaper would be to study how several shapers could be implemented in a single ATM adapter or policing unit to shape several VCs. In this case, there will be some mutual influence among the different shapers.

[2]In this case, the shaper would keep track of average arrival rate of CLP=0 cells, the number of CLP=0 cells in the buffer and use these values to compute the transmission rate for the CLP=0 cells.

However, at the time of this writing, it is still unclear whether the GFR.2 conformance definition will become widely supported by vendors of ATM equipment and whether it will be widely deployed. In fact, partially based on the simulations we described in section 7.3, several members of the ATM Forum have proposed to remove the GFR.2 conformance definition from the GFR service category [HLG97]. Due to administrative reasons, no vote was carried on this proposal and the GFR.2 conformance definition was retained. Although the GFR.2 conformance definition will be part of the GFR service category definition, several influential members of the ATM Forum seem to strongly prefer the GFR.1 conformance definition ...

8.3 Alternative switch implementation

Although the Double-EPD switch implementation provides a lower performance than the per-VC threshold and scheduling implementation when carrying TCP/IP traffic, it has a much lower complexity. The high complexity of the per-VC threshold and scheduling is mainly caused by the utilization of per-VC queues which are served by a WFQ-like scheduler. In addition to their implementation complexity, we have shown that another drawback of the WFQ-like schedulers is that they have difficulties to efficiently support GFR VCs with and without a minimum guaranteed bandwidth.

We investigate in this section whether it is possible to completely support the GFR service category (i.e. both the GFR.1 and the GFR.2 conformance definitions) in FIFO-based switches which do not support per-VC queueing. For this, we discuss an alternative switch implementation. which has been designed with the following goals in mind :

- it should be suitable for FIFO-based switches and should not mandate per-VC queueing ;
- it should completely support the GFR service category (i.e. both the GFR.1 and the GFR.2 conformance definitions) ;
- it should efficiently support GFR VCs with and without a minimum guaranteed bandwidth ;

In this section, we first analyze the limitations of FIFO-based switches when supporting the GFR service category in section 8.3.1. We then describe in sections 8.3.2 and 8.3.3 an alternative switch implementation which fulfills the three goals mentioned above. We then study the performance of this implementation with workstation and internetwork traffic in sections 8.3.4 and 8.3.5. Finally, we compare our alternative implementation with other FIFO-based implementations proposed in the literature in section 8.3.6.

8.3.1 Limitations of FIFO-based switches

To evaluate the limitations of the FIFO-based switches to support the GFR service category, let us consider N GFR VCs multiplexed in a simple FIFO buffer attached to an output link with a bandwidth equal to PCR_o as shown in figure 8.7. In such a FIFO buffer, all the cells which enter the buffer will eventually leave the buffer and be transmitted on the output link. This implies that in a FIFO buffer, the utilization of the output link by the different VCs cannot be controlled directly. The only way to indirectly control the utilization of the output link by the different VCs is to control the arrival[3] of the cells from these VCs in the buffer. This control will be implemented as a buffer acceptance algorithm which decides for each arriving cell whether this cell can be accepted in the FIFO buffer or should be discarded. This is the major limitation of FIFO-based switches compared to switches with per-VC queueing and scheduling. In these latter switches, it is possible to control the utilization of the output link by the different VCs by delaying the transmission of some cells, but without necessarily discarding cells. Due to their intrinsic need to discard cells to control the utilization of their output link by the active VCs, it can be expected that the FIFO-based switches will not be able to achieve the same level of performance as the switches based on per-VC queueing and scheduling.

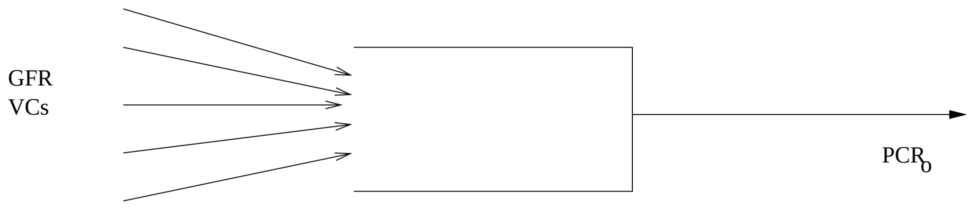

Figure 8.7: Simple FIFO buffer

A simple solution to provide a minimum guaranteed bandwidth to GFR VCs with a single FIFO buffer is to implicitly reserve[4] some buffer space ($Res[i]$) for each VC in proportion to its MCR as shown in equation 8.2. Then, the buffer acceptance algorithm would accept an arriving cell if there are less than $Res[i]$ cells from the corresponding VC in the FIFO buffer.

$$Res[i] = (\frac{MCR[i]}{\sum_{j=1}^{N} MCR[j]}) \times B \tag{8.2}$$

For example, if we consider a FIFO buffer of 1000 cells and two GFR VCs, VC_A with $MCR_A = PCR_o/2$ and VC_B with $MCR_B = PCR_o/3$, then 600 cells are reserved for VC_A and 400 cells for VC_B. If the buffer is always full, then 600 (resp. 400) cells from VC_A (resp. VC_B) are transmitted every 1000 cell slots and thus VC_A (resp. VC_B) achieves a throughput of $0.6 \times PCR_o$ (resp. $0.4 \times PCR_o$). In this case, VC_A and VC_B are able to utilize their minimum guaranteed bandwidth and the unreserved bandwidth ($PCR_o/6$ in our example) is distributed between VC_A and VC_B in proportion to their MCR.

Unfortunately, this solution is not sufficient to completely support the GFR service category when the GFR.1 conformance definition is used. The first problem is that the GFR service category expects the ATM switches to discard entire AAL5-PDUs when they are congested and not to discard individual cells.

[3]Another possible solution would be to perform some explicit congestion notification, e.g. by setting the EFCI bit of some cells and to rely on a specific congestion control mechanism implemented in the endsystems. We do not consider this solution here since the utilization of this bit is not currently part of the GFR service category definition.

[4]We assume, of course, that the CAC algorithm ensures that $\sum_{j=1}^{N} MCR[j] \leq PCR_o$.

A possible solution to this problem is to implicitly divide the buffer in two parts as with the EPD buffer acceptance algorithm. The first part of the buffer (E cells) is implicitly distributed among the established GFR VCs ($Res_E[i]$) in proportion to their MCR according to equation 8.3. When the first cell of an AAL5-PDU arrives, the entire AAL5-PDU is accepted if there are less than $Res_E[i]$ cells from VC[i] in the FIFO buffer. Otherwise the entire AAL5-PDU is discarded. Since an arriving AAL5-PDU is accepted in the buffer without an indication of its size, VC[i] might temporarily utilize more than $Res_E[i]$ cells in the buffer. The second part of the buffer ($B - E$ cells) is used to absorb these temporary overloads due to the tails of accepted AAL5-PDUs. The E value should be chosen to avoid buffer overflow, but should be close to the size of the buffer to allow a good utilization of this buffer.

$$Res_E[i] = \left(\frac{MCR[i]}{\sum_{j=1}^{N} MCR[j]}\right) \times E \tag{8.3}$$

Unfortunately, this solution is still not sufficient to support GFR.1 VCs with a low MCR or a large MBS. To understand the problem caused by such VCs, let us consider a simple example. We assume a 1200 cells long FIFO buffer and set the E threshold to 1000 cells. We consider that two greedy VCs are using this buffer, VC_A with $MCR_A = PCR/100$ and VC_B with $MCR_B = 99 \times PCR/100$ where PCR is the bandwidth of the output link of the FIFO buffer. According to equation 8.3, 10 cells are reserved for VC_A and 990 cells are reserved for VC_B. We also assume that VC_A is sending 192 cells-long AAL5-PDUs (the default MFS for IP over ATM) and for simplicity that VC_B is sending one cell-long AAL5-PDUs.

Let us also suppose that VC_B has been active for some time and that at time $t = 0$ when VC_A becomes active, there are already 990 cells from VC_B in the buffer. The simple example can be easily understood by looking at figure 8.8 which shows the evolution of the buffer occupancy for VC_A and VC_B with time. In this figure, each arrow represents the arrival of the first cell of one AAL5-PDU from VC_A. At time $t = 0$, the first AAL5-PDU from VC_A is accepted since there are no cells from this VC in the buffer. When the first cell of this AAL5-PDU enters the FIFO buffer, there are already 990 cells from VC_B in the FIFO buffer. The occupancy of the FIFO buffer increases between $t = 0$ and $t = 191$ (cell slots) due to this AAL5-PDU. At time $t = 192$ (cell slots), the second AAL5-PDU from VC_A is rejected since there are already more than 10 cells from VC_A in the buffer. The third, fourth, fifth and sixth AAL5-PDUs from VC_A are rejected for the same reason.

A time $t = 990$, the first cell from VC_A is transmitted out of the FIFO buffer and the buffer occupancy starts to decrease. Since VC_A and VC_B are always transmitting cells, cells from VC_A and VC_B are interleaved in the buffer, one cell from VC_A is transmitted out of the FIFO buffer every two cell slots. When the first cell from the seventh AAL5-PDU from VC_A arrives ($t = 1152$), there are still 111 cells from VC_A in the buffer and this AAL5-PDU is rejected. At time $t = 1344$, the eighth AAL5-PDU from VC_A is also rejected since there are still 15 cells from this VC in the buffer. At time $t = 1373$, the last cell from the first AAL5-PDU from VC_A is transmitted out of the FIFO buffer. When the nineth AAL5-PDU from VC_A arrives, there is no cell from this VC in the buffer and the nineth AAL5-PDU is accepted. Thus, on average, one AAL5-PDU out of 8 from VC_A is accepted in the FIFO buffer. This means that VC_A utilizes 1/8 instead of 1/100 of

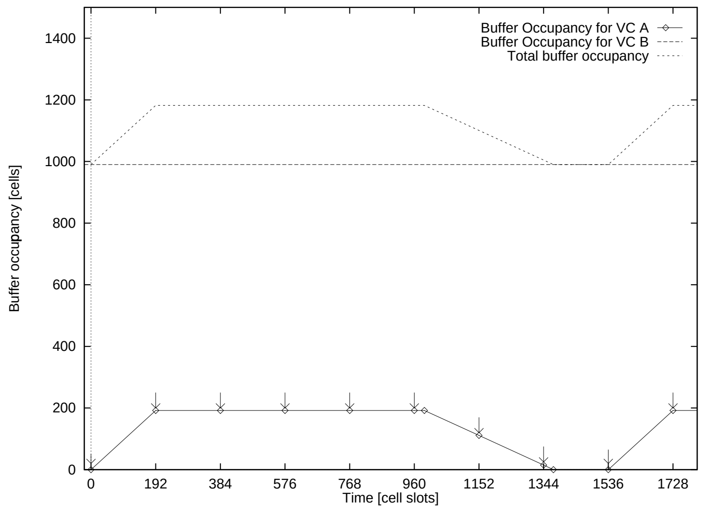

Figure 8.8: Evolution of the occupancy of the FIFO buffer for the simple example

the output link. This is at the detriment of VC_B which is only able to utilize 7/8 of the output link instead of 99/100.

8.3.2 A token-based buffer allocation algorithm (TBA)

To completely support the GFR service category, including GFR.1 VCs with a low MCR and a large MBS, we use a different approach. We first note that the F-GCRA (figure 3.7 on page 43) can be equivalently[5] written as a token bucket as shown in figure 8.9. When the VC is established, the bucket is full and contains MBS tokens. When the first cell of an AAL5-PDU arrives, the entire AAL5-PDU is declared eligible if there is at least one token in the bucket and one token is removed from the bucket for each eligible cell. It should be noted that the number of tokens in the bucket can become negative since the content of the bucket is only checked upon the arrival of the first cell of an AAL5-PDU. When the bucket is not full, one token is placed in the bucket every $1/MCR$ seconds with a timer-based mechanism.

We use this equivalent F-GCRA as the basis for our token-based buffer allocation algorithm (TBA) by applying the virtual queueing principle proposed in [CXK97] for an ABR switch[6]. Instead of using per-VC counters to count the number of cells from each VC in the FIFO buffer as in several other FIFO-based switch implementations [EA98, GJFV98], we implement a mechanism similar to the token bucket based F-GCRA for each VC and use the per-VC counters to store the number of tokens contained in the bucket of this VC.

For simplicity, we assume an output-buffered switch in this section, but the is not

[5]Figure 8.9 assumes that the timer expiration does not occur while the eligibility of an arriving cell is being checked. Otherwise, the access to the `tokens` variable and the `set_timer` functions must be protected by mechanisms such as semaphores to avoid race conditions between the process which is handling the timer and the process which is checking the eligibility of the arriving cells.

[6]Another GFR switch implementation based on this principle was proposed in [SWR97]. We compare our TBA algorithm with the implementation proposed in [SWR97] in section 8.3.6.

```
Initialization :
    tokens=MBS;

Cell Arrival at time $t_a$ :

First cell of an AAL5-PDU:
if IsCLP1(cell) OR ( tokens$\leq$0)
    /* non-eligible cell */
    eligible=FALSE;
else
{ /* eligible cell */
    eligible = TRUE;
    if(tokens=MBS)
        set_timer (now+1/MCR);
    tokens=tokens-1;
}

Middle cell of an AAL5-PDU:
if (eligible)
    /* eligible cell */
    tokens=tokens-1;
else
    /* non eligible cell */

Timer expiration :
    tokens=tokens+1;
    if (tokens<MBS)
        set_timer(now+1/MCR);
```

Figure 8.9: An equivalent token bucket based F-GCRA

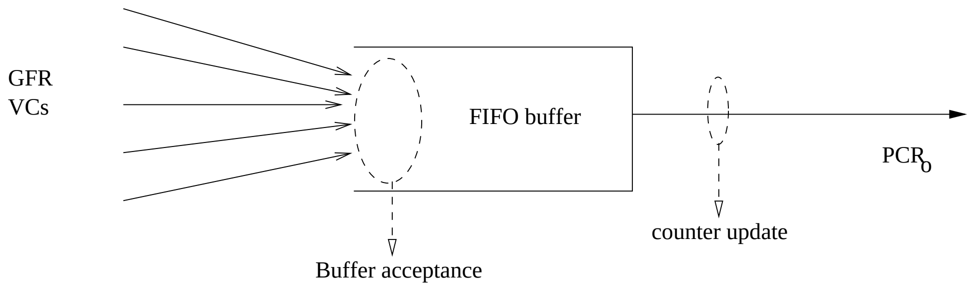

Figure 8.10: Structure of the TBA switch implemenation

restricted to such switches. Our TBA switch implementation is composed of two functional blocks : the buffer acceptance and the counter update algorithms (figure 8.10). The buffer acceptance algorithm decides whether an arriving AAL5-PDU should be accepted in the buffer or not based on the contents of the bucket of the corresponding VC. The counter update algorithm is a background process which replenishes the buckets of the active VCs.

The buffer acceptance algorithm

The buffer acceptance algorithm (figure 8.11) is a direct adaptation of the token bucket based F-GCRA. For simplicity, we assume that only CLP=0 AAL5-PDUs are used on the GFR VCs and defer the discussion on the treatment of CLP=1 AAL5-PDUs until section 8.3.3. The number of tokens in the bucket of VC[i] in the equivalent F-GCRA is stored in a per-VC counter which we call the C[i] counter. When a VC is established, the value of its C[i] counter is set to the MBS of its traffic contract. An arriving AAL5-PDU is accepted in the FIFO-buffer if the value of its C[i] counter is positive. Otherwise, the entire AAL5-PDU is discarded. Furthermore, the C[i] counter is decremented by one every time a cell of the corresponding VC is accepted in the FIFO buffer.

```
Arrival of the first cell from an AAL5-PDU on VC[i] :

if(C[i] > 0)
  {
    accept_PDU[i] = TRUE; /* entire AAL5-PDU is accepted */
    C[i]=C[i]-1;
    /* Cell is accepted in the buffer */
  }
else
  {
    accept_PDU[i] = FALSE; /* entire AAL5-PDU is rejected */
    discard_cell;
    /* Cell is not accepted in the buffer */
  }

Arrival of a non-first cell from an AAL5-PDU on VC[i] :

if(accept_PDU[i] = TRUE)
  {
    C[i]=C[i]-1;
    /* Cell is accepted in the buffer */
  }
else
  {
    discard_cell;
    /* Cell is not accepted inside the buffer */
  }
```

Figure 8.11: Simple buffer acceptance algorithm

A PPD or EPD mechanism could easily be added to the buffer acceptance algorithm to reduce the impact of or to avoid buffer overflows. These mechanisms do not significantly influence our buffer acceptance algorithm and we do not consider them in this section for simplicity.

From an implementation point of view, the buffer acceptance algorithm can be easily supported in hardware and its memory requirements are modest. The C[i] counters are signed integers with bounded values. In addition to the C[i] counters, the buffer acceptance algorithm requires two one bit flags for each VC (accept_PDU[i] and another one bit flag to identify the beginning of the AAL5-PDUs).

The counter update algorithm

In the equivalent F-GCRA, one timer is associated with each VC. A complete implementation of this F-GCRA (e.g. in a policing unit) would have to maintain one timer for each VC. In a switch which usually needs to support several tens of thousands of VCs on each port, maintaining one explicit timer for each VC would be a very complex solution. Instead of maintaining these timers, we propose an approximate solution. We note that during a period of k cell slots ($\frac{k}{PCR_o}$ seconds), the counter of VC[i] should be incremented $k \times \frac{MCR[i]}{PCR_o}$ times. Instead of associating one timer with each VC, we propose to utilize a predefined schedule which indicates for each cell slot the C[i] counter which must be incremented. This schedule is implemented as a register with W_R positions (the W_R register). Each position in the W_R register is either unassigned or associated with one established VC. An example W_R register with ten positions is shown in figure 8.12. In this example register, five positions are reserved for VC_A, two for VC_B, one for VC_C and two positions are unassigned.

A	B	A	C	A	-	A	B	A	-

Figure 8.12: Example W_R register

This register can be used to increment the counters of the GFR VCs by incrementing one counter every cell slot in the order specified by the W_R register. For example, if we consider the example register shown in figure 8.12, C[A] would be incremented during the first cell slot, then C[B] during the second, C[A] during the third, ..., no counter during the tenth, C[A] during the eleventh, ...

The W_R register can be used as a schedule to update the C[i] counters provided that $W_R \times \frac{MCR[i]}{PCR_o}$ positions are allocated to VC[i]. Since the number of positions for each VC in the W_R register is a function of the MCR of each VC, we have to modify the W_R register every time a new VC is established or released. However, since the number of positions for one VC does not depend on the characteristics of the other VCs, the W_R register can be incrementally updated every time a new VC is established or released. When a VC is released, the positions assigned to this VC can simply be marked as unassigned. When a new VC is established, unassigned positions can be assigned to this VC with the incremental algorithm shown in figure 8.13. This algorithm assigns $\lceil (W_R \times \frac{MCR[i]}{PCR_o}) \rceil$ positions for VC[i] and tries to evenly distribute the positions assigned to this VC in the W_R register. Other algorithms are possible.

Since each established GFR VC occupies several positions in the W_R register, a minimum condition to accept the establishment of a new VC is that the W_R register does not

```
Update of $W_R$ register to include VC[i] :

L[i] = $PCR_o$/MCR[i]; /* ideal distance in $W_R$ */
N[i] = $\lceil (W_R \times MCR[i])/PCR_o \rceil$; /* # positions in $W_R$ */
I_pos= random(0,L[i]); /* 1st pos.  for VC[i] 0$\leq$ I_pos $<$ L[i]*/
for (j=0; j<N[i] ; j++)
  {
    ideal_pos = floor(I_pos+j $\times$ L[i]);
    if ($W_R$[ideal_pos] is free)
      reserve position ideal_pos for VC[i];
    else
      {
        backtrack=1;
        while ((ideal_pos-backtrack $\geq$ 0) OR
                (ideal_pos+backtrack< $W_R$) )
          {
            if (ideal_pos-backtrack $\geq$ 0) AND ($W_R$[ideal_pos-backtrack] is free)
              {
                reserve position ideal_pos-backtrack for VC[i];
                backtrack=$W_R$; /* leave loop */
              }
            else
              if (ideal_pos+backtrack< $W_R$) AND ($W_R$[ideal_pos+backtrack] is free)
                {
                  reserve position ideal_pos+backtrack for VC[i];
                  backtrack=$W_R$; /* leave loop */
                }
            backtrack=backtrack+1;
          }
      }
  }
```

Figure 8.13: Incremental update of the W_R register

become full. A complete Connection Admission Control (CAC) algorithm would require additional conditions, notably a relation between the MBS of the established VCs and the total amount of available buffer. In a real implementation, the size of the W_R register will be limited. This implies that the smallest MCR which can be supported by the TBA switch implementation is equal to PCR_o/W_R. Thus a large W_R register is required to support GFR VCs with a low MCR.

The counter update algorithm examines one position of the W_R register every time a cell is sent out of the FIFO buffer as shown in figure 8.14. If this position is assigned to a GFR VC, the C[i] counter of this VC is incremented provided that its value is not already equal to the MBS of the traffic contract of this VC. In figure 8.14, IsGFRVC(W_R[k]) is true when the k^{th} position in the W_R register is assigned to a GFR VC and C[W_R[j]] is used to indicate the counter which corresponds to the VC found in the j^{th} position of the W_R register.

```
Cell transmission from FIFO buffer:

  j=(current_pos+1) mod $W_R$;
  if( IsGFRVC($W_R$[j]) and C[$W_R$[j]] < MBS[$W_R$[j]] )
      C[$W_R$[j]]=C[$W_R$[j]]+1; /*incr. counter of active VC*/
  current_pos=j;
```

Figure 8.14: Counter update algorithm : provision of the MCR guarantee

The buffer acceptance algorithm shown in figure 8.11 combined with the counter update algorithm shown in figure 8.14 accepts only the AAL5-PDUs which are eligible for the minimum guaranteed bandwidth in the FIFO buffer. This is not a very efficient implementation since in this case the GFR VCs cannot utilize more than their reserved bandwidth and thus cannot completely utilize the available bandwidth.

To allow the active VCs to benefit from any available bandwidth in addition to their minimum guaranteed bandwidth, we note that when the W_R register points to an unassigned position or to an inactive VC (i.e. a VCwith a bucket which already contains MBS tokens), then one token can be safely provided to another VC without impacting the provision of the tokens to the active VCs to fulfill their MCR. We will call these tokens the "excess" tokens in the remainder of this section. Over some period of time, the number of excess tokens is an image of the amount of bandwidth which can be safely used by the active VCs without jeopardizing the minimum guaranteed bandwidth of the other VCs. Since many VCs could potentially benefit from these excess tokens, we must define a strategy to distribute these tokens.

A first strategy could be to distribute, over a long period, the same number of excess tokens to each active VC. With this strategy, the available unreserved and unused reserved bandwidth is shared equally among all the active VCs. A second strategy could be to distribute the excess tokens in proportion to the MCR of the active VCs[7]. In this case, the TBA switch implementation would distribute the available bandwidth like the per-VC threshold and scheduling implementation. A third strategy could be to define two classes of VCs. The "local" class would correspond to the VCs which originate from the local ATM network and thus are directly charged by the network operator for their utilization of the bandwidth. The "transit" class would correspond to transit VCs which originate from a different ATM network. If the network operator receives more revenue from the VCs in the local class, it could want to reward them by allowing these local VCs to utilize a larger fraction of the unreserved bandwidth than the transit VCs.

Many different strategies to distribute the excess tokens are possible. We expect that network operators would prefer to be able to select the strategy which better suits

[7]We have described in [Bon98a] a counter update algorithm which supports this strategy by using only the W_R register. This counter update algorithm uses the W_R register in the same way as a Weighted Round-Robin (WRR) scheduler.

their business needs instead of being forced to use the strategy chosen by the switch manufacturer.

To support these different strategies with a single counter update algorithm, we propose to associate one administrative weight (Q[i]) with each VC. This administrative weight is used to determine how the excess tokens are distributed among the active VCs. We propose to distribute the excess tokens to the active VCs according to equation 8.4 where U is the total number of excess tokens.

$$Excess[i] = U \times \frac{Q[i]}{\sum_{j=Active\ VCs} Q[j]} \tag{8.4}$$

The utilization of this equation allows the network operator to support different strategies for the allocation of the excess tokens by an appropriate choice of the administrative weights. For example, if a network operator wants to equally distribute the excess tokens to the active VCs, then all the administrative weights would be set to the same value (e.g. 1). If a second network operator wants to distribute the excess tokens in proportion to the MCR of the active VCs, the administrative weights should be equal to the MCRs. If a third operator wants to provide tens times more excess tokens to its local VCs than to the transit VCs, the administrative weights of the local VCs should be ten times larger than the administrative weights of the transit VCs.

To distribute the excess tokens according to equation 8.4, we use a second register (the W_U register). Like the W_R register, the W_U register is also used to indicate the counters which need to be updated. To populate the W_U register, we use the algorithm shown in figure 8.13, with Q[i] instead MCR[i]. Thus, VC[i] will occupy $\frac{Q[i]}{W_U}$ positions in the W_U register.

The counter update algorithm which utilizes the two registers is shown in figure 8.15. Intuitively, the modified counter update algorithm works as follows. One token is provided to one VC every time a cell is sent out of the FIFO buffer. The bucket of the VC corresponding to the current position in the W_R register is first checked and if required a token is supplied to this VC. Otherwise, the counter update algorithm finds the first active VC in the W_U register (from the current position in the W_U register onwards) and supplies one token to the found VC. It is easy to see that the algorithm described in figure 8.15 distributes the excess tokens according to equation 8.4 when considering greedy VCs. In fact, the W_U register is used like a Weighted Round-Robin (WRR) scheduler [KSC91]. We expect that in a real implementation, the comparisons C[W_R[j]] < MBS[W_R[j]] and C[W_U[j]] < MBS[W_U[j]] shown in figure 8.15 will be avoided by using a one-bit flag to indicate whether the bucket of VC[i] is full or not.

An important property of the counter update and buffer acceptance algorithms is that the values of the C[i] counters are bounded. First, as the counter update algorithm does not increment the counter of the inactive VCs, the value of the C[i] counters cannot become larger than MBS[i]. Second, since one counter is decremented by one when a cell arrives and one counter is incremented by one every time a cell leaves the buffer, the values of the C[i] counters and the total occupancy of the buffer ($B_{Occupancy}$) are linked by equation 8.5. This implies that the counter update algorithm does not need to be invoked when the buffer is empty, since all the VCs are inactive at that time.

```
Cell transmission from FIFO buffer:
/* assumes that there is at least one cell in the buffer */
current_R=(current_R+1) mod $W_R$;
if (IsGFRVC($W_R$[current_R]) and C[$W_R$[current_R]]< MBS[$W_R$[current_R]])
    /* VC pointed in $W_R$ is active */
    C[$W_R$[current_R]]= C[$W_R$[current_R]]+1;
else
  {
    /* find first active VC in $W_U$ register */
    j=(current_U+1) mod $W_U$;
    /* find first active VC in $W_U$*/
    while not( IsGFRVC($W_U$[j]) and C[$W_U$[j]] < MBS[$W_U$[j]])
        j=(j+1) mod $W_U$;
    C[$W_U$[j]]=C[$W_U$[j]]+1; /* increment counter of found VC */
    current_U=j;
  }
```

Figure 8.15: Counter update algorithm with allocation of the excess tokens

$$\sum_j C[j] + B_{Occupancy} = \sum_j MBS[j] \tag{8.5}$$

An important point to note concerning the counter update algorithm is that it is activated every time a cell is transmitted, but the counter which is incremented is not necessarily the counter of the VC whose cell is currently transmitted (e.g. the C[i] counter of VC 123 could be incremented while a cell from VC 456 is being transmitted). The only requirement on the counter update algorithm is that one C[i] counter is found and incremented within one cell slot.

8.3.3 Enhancements to the simple TBA switch implementation

The simple TBA switch implementation discussed in the previous section is not yet sufficient to completely support the GFR service category, since it does not distinguish between CLP=1 and CLP=0 AAL5-PDUs. This distinction is required to completely support the GFR.1 and GFR.2 conformance definitions. Before explaining how the simple TBA switch implementation should be modified to support these two conformance definitions, we first describe an important improvement to the buffer acceptance algorithm to better support TCP/IP traffic by controlling the buffer occupancy. We then explain how to also support nrt-VBR and UBR VCs in the same FIFO buffer as the GFR VCs and discuss the integration of our TBA switch implementation in multi-service switches.

Control of the buffer occupancy

In a real ATM switch, there are many practical reasons to try to maintain the average buffer occupancy at a low value while still providing the required QoS guarantees. If the buffer is always completely filled, then the link utilization is good, but the average delay through the buffer is large and furthermore the buffer may have difficulties to correctly handle bursty traffic without losses. For the GFR service category, a buffer which is always filled would probably have difficulties to provide the MBS part of the minimum guaranteed bandwidth for all the GFR VCs. On the other hand, if the buffer is often empty, then the average delay through the buffer is small and bursty traffic can be easily handled without losses, but the link utilization may be very low.

Preliminary simulations performed with workstation and internetwork traffic with the simple TBA switch implementation discussed in the previous section have shown that TCP had difficulties to efficiently utilize the minimum guaranteed bandwidth of the underlying ATM VCs. The main reason for this low performance was that the switch buffers were often empty and thus the NNI links were not 100% utilized, especially with a low MBS for the GFR traffic contracts. The main reason for the low occupancy of the switch buffers is that the buffer acceptance algorithm shown in figure 8.11 is rigid and only considers the value of the C[i] to decide whether an arriving AAL5-PDU should be accepted or not. This means that AAL5-PDUs are discarded even when the buffer occupancy is very small. In this case, a better solution would be to have a less rigid buffer acceptance algorithm which may still accept AAL5-PDUs on VCs with an empty or even negative token bucket provided that the occupancy of the switch buffer is low.

As a basis for a less rigid buffer acceptance mechanism, we consider the Random Early Detection (RED) algorithm which was proposed in [FJ93] for IP routers and is considered to be particularly efficient to support TCP/IP traffic [BCC$^+$98].

The RED algorithm controls the arrival of IP packets into a buffer and tries to maintain the average occupancy of this buffer between two predefined thresholds ($min_{th} < max_{th}$). For this, it continuously measures the average occupancy (avg) of its buffer and decides to accept or reject an arriving IP packet based on its average buffer occupancy. If the average buffer occupancy is low ($avg < min_{th}$), the arriving packet is accepted. If the average buffer occupancy is between the predefined thresholds ($min_{th} \leq avg < max_{th}$), the arriving packet is probabilistically accepted or discarded. The packet drop probability is computed as a linear function of the average buffer occupancy and this function is equal to max_p when the average buffer occupancy is close to the higher threshold. If the average buffer occupancy is too large ($avg > max_{th}$), the arriving packet is discarded.

To adapt the RED algorithm to our TBA switch implementation, we first need to measure the average buffer occupancy. This can be done by computing this average buffer occupancy every cell slot as $avg = avg + wq \times (B_{occupancy} - avg)$ as proposed in [FJ93]. We introduce some flexibility in the buffer acceptance algorithm by applying a slightly modified RED algorithm when the C[i] counter corresponding to the arriving AAL5-PDU is negative (figure 8.16). When the C[i] counter is positive, the arriving eligible AAL5-PDU is automatically accepted in order to fulfill the minimum guaranteed bandwidth of VC[i]. In this modified buffer acceptance algorithm, we consider three different cases when the value of the C[i] counter is negative :

- The buffer is lightly filled ($avg < min_{th}$). In this case, the arriving AAL5-PDU is accepted since there is no risk of congestion.

- The buffer is correctly filled ($min_{th} \leq avg < max_{th}$). In this case, the arriving AAL5-PDU should be probabilistically accepted. Instead of computing the packet dropping probability solely on the basis of the average buffer occupancy as in the RED algorithm, we also rely on the value of the C[i] counter[8]. Intuitively, we would like to have a high packet drop probability for a VC with a negative C[i] counter with a large absolute value and a lower packet drop probability for a VC with a negative but close to 0 C[i] counter since the former VC has caused more congestion than the latter VC. We choose a packet drop probability which is a linear function of the average buffer occupancy and the C[i] counter (figure 8.16). The relative influence of the average buffer occupancy and the C[i] counter on the packet dropping probability depend on the values of α and β. Intuitively, α is similar to max_p in the original RED algorithm. α is the packet drop probability for a VC with a C[i] counter close to zero when the average buffer occupancy is close to max_{th}. β is the packet drop probability for a VC whose C[i] counter is equal to $-max_{th}$ when the average buffer occupancy is equal to min_{th}. In practice, we expect that the value of α will be much smaller than the recommended values for max_p in the original RED algorithm and that β will be larger than α. This choice of parameters will ensure that the VC which is responsible for the congestion will be mainly affected by the random losses.

- The buffer is heavily used ($max_{th} < avg$). In this case, the arriving AAL5-PDU is discarded.

The complete modified buffer acceptance algorithm is shown in figure 8.16. The solution proposed in [FJ93] to generate only one random number per dropped packet can also be applied to the modified buffer acceptance algorithm.

Support for GFR.1 and GFR.2

To completely support the GFR.1 and GFR.2 conformance definitions, our TBA switch implementation should provide a different treatment for the CLP=0 and the CLP=1 AAL5-PDUs. When the GFR.2 conformance definition is used, the switch should not discard the CLP=0 AAL5-PDUs. This can be done by always accepting the CLP=0 AAL5-PDUs in the buffer, independently of the value of the C[i] counter. For the CLP=1 AAL5-PDUs, we propose to accept them probabilistically as shown in figure 8.17.

When considering the support of the GFR.1 VCs, we have to distinguish between the two interpretations for the CLP bit identified in section 7.5.2. To support the relative CLP priority, we propose the modifications to the buffer acceptance algorithm shown in figure 8.18. With this buffer acceptance algorithm, an arriving CLP=0 AAL5-PDU is accepted independently of the average buffer occupancy if the corresponding C[i] counter is positive. An arriving CLP=1 AAL5-PDU is accepted independently of the average buffer occupancy only if its C[i] counter is larger than some positive value (e.g. MBS[i]/2). When an arriving AAL5-PDU must be probabilistically accepted or rejected, we propose to have

[8]To follow the "RED principle", the buffer acceptance algorithm should ideally maintain one "average" C[i] counter and use this average value to compute the packet drop probability. However, maintaining these additional per-VC "average" C[i] counter would significantly increase the per-VC memory requirements of our TBA switch implementation. To avoid this additional memory requirement, we do not maintain an average value for the C[i] counters and only use their instantaneous values. It should however be noted that our modified buffer acceptance algorithm still uses, like RED, the average buffer occupancy as an indication of congestion.

```
Arrival of the first cell from an AAL5-PDU on VC[i] :

if(C[i]$>$ 0)
{
  accept_PDU[i] = TRUE; /* entire AAL5-PDU will be accepted */
}
else
  if (avg$<$ $min_{th}$)
  { /* Buffer is lightly filled */
    accept_PDU[i]=TRUE;
  }
  else
    if (avg$<$ $max_{th}$)
      { /* Buffer is correctly occupied thus random drop */
        Pb= $\alpha\times$(avg-$min_{th}$)/($max_{th} - min_{th}$) - C[i]$\times\beta/max_{th}$;
        With probability Pb
          /* entire AAL5-PDU will be discarded */
          accept_PDU[i]=FALSE;
        else
          /* entire AAL5-PDU will be accepted */
          accept_PDU[i]=TRUE;
      }
      else
      { /* Buffer is heavily occupied */
        accept_PDU[i]=FALSE;
      }

if(accept_PDU[i]=TRUE)
{
  C[i]=C[i]-1; /* decrement counter */
  /* Cell is accepted in the buffer */
}
else
{
  discard_cell;
  /* Cell is not accepted in the buffer */
}
```

Figure 8.16: Modified buffer acceptance algorithm

a larger packet drop probability for CLP=1 AAL5-PDU by using a larger β parameter for the CLP=1 AAL5-PDUs (e.g. by using $\beta_1 = 2 \times \beta$). With this modified buffer acceptance algorithm, the discard probability for CLP=1 AAL5-PDUs will be higher than for CLP=0 AAL5-PDUs.

To support the strict interpretation for the CLP bit, we propose to use the technique proposed in [CW97b] for IP routers. With a strict interpretation for the CLP bit,

```
Arrival of the first cell from an AAL5-PDU on (GFR.2) VC[i] :

if (IsCLP0(cell))
  accept_PDU[i] = TRUE; /* entire AAL5-PDU will be accepted */
else if (avg< $min_{th}$)
  { /* Buffer is lightly filled, AAL5-PDU is accepted */
    accept_PDU[i]=TRUE;
  }
  else if (avg< $max_{th}$)
  { /* Buffer is correctly occupied thus random drop */
    Pb=$\alpha \times (avg - min_{th})/(max_{th} - min_{th}) - min(0, C[i]) \times \beta/max_{th}$;
    ...
```

Figure 8.17: Modified buffer acceptance algorithm for GFR.2 VCs

```
Arrival of the first cell from an AAL5-PDU on (GFR.1) VC[i] :

if ((IsCLP0(cell) AND (C[i] > 0) OR
    (IsCLP1(cell) AND (C[i]>MBS[i]/2)) )
  accept_PDU[i] = TRUE; /* entire AAL5-PDU will be accepted */
...
  else if (avg< $max_{th}$)
  { /* Buffer is correctly occupied thus random drop */
    if (IsCLP0(cell))
      Pb=$\alpha \times (avg - min_{th})/(max_{th} - min_{th}) - C[i] \times \beta/max_{th}$;
    else
      Pb= $\alpha \times (avg - min_{th})/(max_{th} - min_{th}) - min(0, C[i]) \times \beta_1/max_{th}$;
    ...
```

Figure 8.18: Modified buffer acceptance algorithm for GFR.1 with relative CLP priority

the CLP=1 AAL5-PDUs should only be accepted when there are not enough CLP=0 AAL5-PDUs to completely utilize the available bandwidth. This can be done with the Random Early Detection with In/Out (RIO) buffer acceptance algorithm proposed in [CW97b]. Two different RED-based algorithms are used. The first one measures the average buffer occupancy for the CLP=0+1 cells and probabilistically discards the arriving CLP=1 AAL5-PDU when this average buffer occupancy is between the min_{th}^{0+1} and max_{th}^{0+1} thresholds. The second RED-based algorithm measures the average buffer occupancy for the CLP=0 cells and probabilistically discards an arriving CLP=0 AAL5-PDU when this average buffer occupancy is between the min_{th}^{0} and max_{th}^{0} thresholds. The four thresholds should be chosen so that $max_{th}^{0+1} \leq min_{th}^{0}$ to ensure that CLP=1 AAL5-PDUs are only accepted when the average buffer occupancy for the CLP=0 cells is low. This implies that CLP=1 AAL5-PDUS will only be accepted when there are not enough CLP=0 AAL5-PDUs to completely utilize the buffer and the available bandwidth.

Support for the real-time service categories

Until now, we have considered an ATM switch which was only supporting the GFR service category. In practice, most ATM switches also need to support VCs from other service categories. The real-time service categories[9] (CBR and rt-VBR.1) usually require tight delay guarantees. For this reason, the traffic from the real-time VCs cannot be multiplexed with the traffic from the GFR VCs in a single FIFO buffer. A common solution to support real-time and non-real-time service categories in one ATM switch is to use one buffer for the non-real-time service categories and another (small) buffer for the real-time service categories. Then, a priority scheduler is used to always serve the real-time buffer when it is not empty and to only serve the non-real-time buffer when the real-time buffer is empty (figure 8.19). When GFR and CBR VCs have to be supported by the same switch, the CAC algorithm must ensure that the bandwidth which is leftover by the real-time VCs is larger than the total reserved bandwidth for all the GFR VCs.

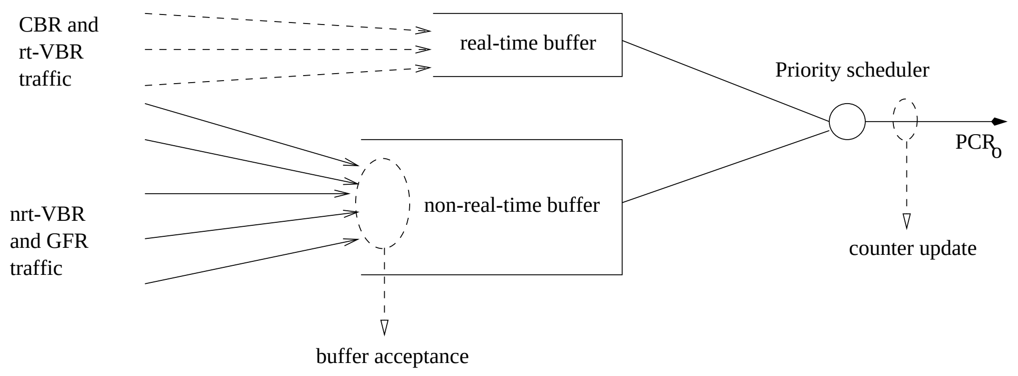

Figure 8.19: Multi-service switch with priority scheduler

When we considered a single FIFO buffer drained at a constant rate PCR_o, the counter update algorithm was able to share the available bandwidth among the active VCs in proportion to their MCR and administrative weights. In a multi-service switch, the bandwidth which is available to the GFR VCs is not anymore constant and equal to PCR_o, but depends on the amount of real-time traffic. In order to distribute this bandwidth among the GFR VCs, the counter update algorithm needs to "measure" the amount of bandwidth which is consumed by the real-time VCs. This can be performed with a small modification to the counter update algorithm. The modified counter update algorithm (figure 8.20) is still activated every time one cell is transmitted on the output link. In addition to the C[i] counters and the W_R register, we use a new global integer variable called "Z" (initialized to zero). "Z" is an indication of the number of "tokens" which have been recently allocated while cells from the real-time buffer were being transmitted.

The modified counter update algorithm works as follows. When a cell is transmitted from the real-time buffer, Z is incremented by one. The algorithm then checks the next position in the W_R register. If this position corresponds to an active VC, the counter of this VC is incremented. This is used to ensure that the C[i] counters of the active VCs

[9]In this section, we do not consider the rt-VBR.2 and rt-VBR.3 conformance definitions because they cannot be efficiently supported by switches without per-VC queueing and scheduling.

are still updated at a rate equal to their MCR even during a burst of real-time traffic. Otherwise the value of Z is checked. If Z is positive, this means that the algorithm has updated some C[i] counters while real-time cells were transmitted. In this case, no C[i] counter is incremented during this cell slot but Z is decremented by one. This is equivalent to virtually allocating a token to the real-time buffer[10] and allows the counter update algorithm to keep track of the bandwidth consumed by the real-time cells. When Z is equal to zero, the number of tokens provided to the GFR VCs is equal to the number of cells sent from the GFR buffer. In this case, the C[i] counter of the first active VC in W_U is incremented.

```
At every cell transmission (from real-time or GFR buffer) :

if (a cell is transmitted from real-time buffer)
    Z=Z+1;
current_R=(current_R+1) mod $W_R$;
if (IsGFRVC($W_R$[current_R]) and C[$W_R$[current_R]]< MBS[$W_R$[current_R]])
    /* VC pointed in $W_R$ is active */
    C[$W_R$[current_R]]= C[$W_R$[current_R]]+1;
else
  {
    if (Z>0)
        /* Buffer is drained at lower rate than $PCR_o$ */
        Z=Z-1;
    else
      { /* Allocate unused bandwidth according to $W_U$ */
        /* find first active VC in $W_U$ register */
        j=(current_U+1) mod $W_U$;
        /* find first active VC */
        while not( filled($W_U$[j]) and C[$W_U$[j]] < MBS[$W_U$[j]])
            j=(j+1) mod $W_U$;
        C[$W_U$[j]]=C[$W_U$[j]]+1; /* increment counter of found VC */
        current_U=j;
      }
  }
```

Figure 8.20: The counter update algorithm for multi-service switches

Support for the nrt-VBR service category

To support (non-real-time) VBR VCs in the same buffer as the GFR VCs, we can slightly modify the buffer acceptance algorithm. The buffer acceptance algorithm for the VBR VCs still relies on the W_R register, but the acceptance decision is on a per cell basis and not on a per AAL5-PDU basis.

[10]In multi-service switches, equation 8.5 becomes $\sum_j C[j] + B_{Occupancy} - Z = \sum_j MBS[j]$, where $B_{Occupancy}$ is the number of cells in the non-real-time buffer.

When a VBR VC is established, we reserve some positions for this VC in the W_R register in proportion to its SCR and possibly some positions in the W_U register. When considering the VBR VCs, we have to distinguish between VBR.1 and VBR.2/VBR.3. With VBR.1, all the cells (CLP=1 and CLP=0) must be accepted inside the buffer since the QoS guarantees apply to the CLP=0+1 cell flow, while for the VBR.2 and VBR.3 VCs the CLP=0 cells must always be accepted while the CLP=1 cells should only be accepted when there is no congestion.

Support for the UBR service category

The last service category[11] that we consider is the UBR service category. The UBR (or GFR with MCR=0) VCs do not have any minimum bandwidth, but they should be able to utilize some fraction of the unreserved bandwidth like the GFR VCs with a non-zero MCR. To support these best-effort VCs which rely on AAL5 like the GFR VCs, but without any minimum guaranteed bandwidth, we still associate one C[i] counter with each best-effort (i.e. UBR or GFR with MCR=0) VC. However, since these VCs do not have any minimum guaranteed bandwidth we do not associate positions of the W_R register to these VCs. However, we associate one administrative weight, and thus at least one position in the W_U register to each best-effort VC. With an appropriate choice for the administrative weights, the network operator can completely control how the available bandwidth is distributed among the active guaranteed and best-effort VCs without requiring an implicit reservation of some amount of bandwidth for the best-effort VCs.

8.3.4 Performance with workstation traffic

To evaluate the suitability of our TBA switch implementation to carry workstation traffic, we performed several simulations with the simple ATM LAN used in section 7.3. We used exactly the same simulation model as in section 7.3, but with our TBA switch implementation instead of the per-VC threshold and scheduling switch implementation. We assume that the workstations are only sending CLP=0 AAL5-PDUs and only consider the GFR.1 conformance definition in this section. In order to compare our TBA switch implementation with the per-VC threshold and scheduling implementation, we set the administrative weights of all the GFR VCs to the MCR of their traffic contract. With these administrative weights, our TBA switch implementation should provide the same distribution of the available bandwidth as the per-VC threshold and scheduling implementation.

For the first simulations with our TBA switch implementation, we considered nine background workstations and set their MCR to 20000 cells per second. We still varied the MCR of the privileged workstation from 20000 to 180000 cells per second. We set α to 0, β to 0.1 and wq to 0.001 and used registers with 100 positions and set the MBS of the GFR traffic contracts to 200 cells. The simulations performed with $TCP_{default}$ and a MSS of 1460 bytes were somewhat disappointing (figure 8.21). With $TCP_{default}$, the privileged workstation was unable to utilize its minimum guaranteed bandwidth and

[11]A support for the closed-loop service categories (e.g. ABR) would require some interactions between the algorithm used to compute the RM cells and the counter update and buffer acceptance algorithm of our TBA switch implementation. This is outside the scope of this thesis.

the performance of our TBA switch implementation was similar to the performance of the much simpler Double-EPD implementation. This is not too surprising since our TBA switch implementation can only control the allocation of the bandwidth to the different VCs by discarding AAL5-PDUs and, as shown in section 7.3.2, $TCP_{default}$ is very sensitive to packet losses. The main reason for the low performance of our TBA switch implementation with $TCP_{default}$ is the large number of expirations of the retransmission timer. For example, during the simulations with the MCR of the privileged workstation set to 180000 cells per second, its retransmission timer expired on average once per second.

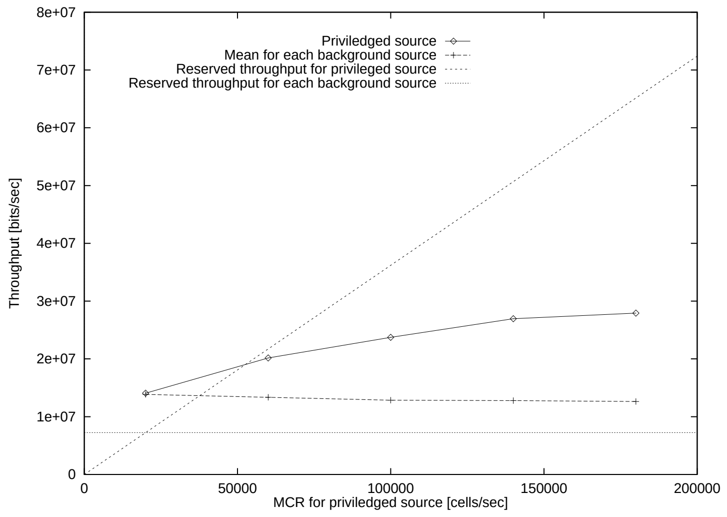

Figure 8.21: $TCP_{default}$ throughput with 1460 bytes MSS and TBA switch implementation

We also performed simulations with other values for α and β as well as with a larger MBS and a smaller MSS. By using a larger MBS, we could slightly reduce the number of expirations of the retransmission timer and thus slightly improve the throughput of the workstations, but the privileged workstation was still unable to utilize its minimum guaranteed bandwidth. The simulations performed with a MSS of 9140 also revealed that the privileged source was unable to utilize its minimum guaranteed bandwidth.

The simulations performed with TCP_{fast} produced slightly better results than $TCP_{default}$, but the privileged source was only able to utilize about 50% of its minimum guaranteed bandwidth when its MCR was set to 180000 cells per second.

The simulations performed with TCP_{SACK} produced better results. With TCP_{SACK}, the privileged workstation was able to better utilize its minimum guaranteed bandwidth. We performed simulations with different values for α, β, the MBS of the GFR traffic contracts and the MSS used by the workstations. For example, in figure 8.22, we show the results of a simulation with a 1460 bytes MSS, $\alpha = 0$, $\beta = 0.1$ and a MBS of 200 cells. The performance of TCP_{SACK} was slightly higher with a MBS of 1000 cells.

The simulations with a MSS of 9140 bytes and TCP_{SACK} showed that with this large MSS, the performance of TCP_{SACK} was very sensitive to the values of the α and β parameters. With some values for these parameters, the privileged source could only utilize a fraction of its minimum guaranteed bandwidth while with other values for these parameters the privileged source could utilize more bandwidth than its minimum guaranteed bandwidth at the detriment of the background sources which could not utilize their

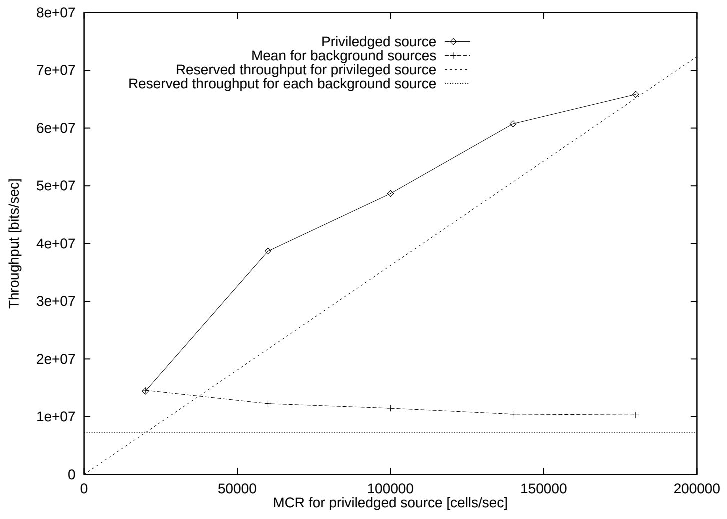

Figure 8.22: TCP_{SACK} throughput with 1460 bytes MSS and TBA switch implementation

minimum guaranteed bandwidth. However, an interesting point to note concerning our TBA switch implementation is that, even with a 9140 bytes MSS, its performance with TCP_{SACK} was better than the performance with the per-VC threshold and scheduling implementation when packet losses occured (figures 7.19 and 7.20). Thus with our TBA switch implementation, TCP_{SACK} seems to be slightly less sensitive to packet losses than with per-VC threshold and scheduling.

When comparing our TBA switch implementation with the Double-EPD and per-VC threshold and scheduling implementations, we have to distinguish between $TCP_{default}$ and TCP_{fast} on one hand and TCP_{SACK} on the other hand. With $TCP_{default}$ and TCP_{fast}, our TBA switch implementation does not bring, from a performance point of view, any significant benefit compared with the much simpler Double-EPD implementation. With TCP_{SACK}, our TBA switch implementation provides a better performance than the Double-EPD implementation. With $TCP_{default}$ and $TCP_{default}$, the per-VC threshold and scheduling implementation appears to be, from a pure performance point of view, the best switch implementation to support workstation traffic, although its buffer acceptance algorithm could probably be improved. From a hardware complexity point of view, our TBA switch implementation is between the Double-EPD and the per-VC threshold and scheduling implementations.

8.3.5 Performance with internetwork traffic

To evaluate the performance of the TBA switch implementation with internetwork traffic, we first consider the “five routers” scenario.

In figure 8.23, we compare the performance of our TBA switch implementation with the per-VC threshold and scheduling implementation. We set in both cases the MBS of the GFR traffic contract equal to the MFS (32 cells). These simulations show that with our TBA switch implementation, the workstations attached to the various routers are able to efficiently utilize their minimum guaranteed bandwidth. An interesting point to note concerning our TBA switch implementation is that its performance in the five routers

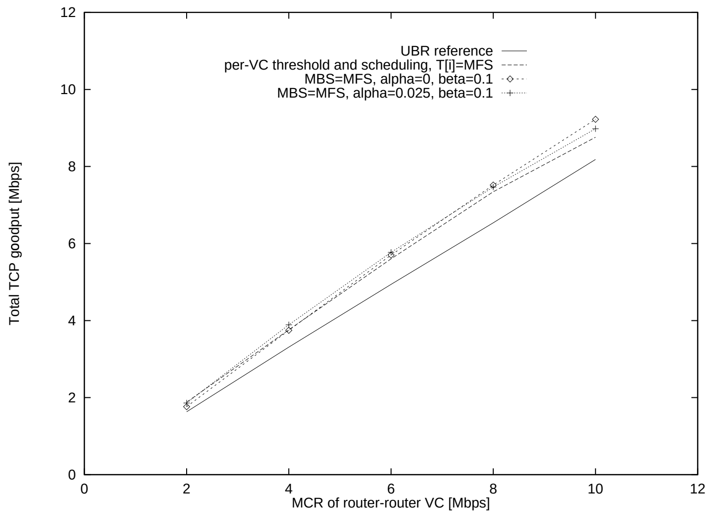

Figure 8.23: Five routers scenario with TBA switch implementation

scenario does not appear to be heavily dependent on the values of the chosen parameters, provided that the values of these parameters are within a reasonable range. Furthermore, the comparison between our TBA switch implementation and the per-VC threshold and scheduling implementation shows a small advantage for our TBA switch implementation. A closer look at the simulation results revealed that the better performance of our TBA switch implementation was due to a lower number of duplicate packets received by the destination workstations. This is probably due to the utilization of probabilistic drops in the buffer acceptance algorithm of our TBA switch implementation.

We also performed simulations with the "ten routers" scenario. These simulations (figure 8.24) showed that with our TBA switch implementation, there was a small unfairness at the detriment of the routers with a large round trip time. However, it should be noted that this unfairness is less severe with our TBA switch implementation than with the Double-EPD implementation. Furthermore, a comparison with the per-VC threshold and scheduling implementation does not reveal a significant difference between the performance of this implementation and our TBA switch implementation.

For the simulations reported in figure 8.24, we set the min_{th} and max_{th} threshold to respectively 2000 and 6000 cells. We achieved good performance with the following choice of parameters : $\alpha = 0$ and $\beta = 0.1$. The performance was slightly better (i.e. the unfairness for the routers with a large rtt was slightly reduced) with a larger MBS, but large values for the MBS are not required to achieve a good performance, unlike the Double-EPD implementation.

With the ten routers scenario, the performance of our TBA switch implementation appeared to be more sensitive to the value of α and β than with the five routers scenario. Furthermore, α seems to have a higher influence than β on the performance of our TBA switch implementation with internetwork traffic. Table 8.1 shows the total goodput achieved by the workstations attached to the 4 Mbps router with a 105 msec delay with the MBS of the GFR traffic contracts equal to the MFS. When α was set to 0, a small value for β produced a lower performance than larger values (e.g. $\beta \geq 0.1$). When β was set to 0.1, a small increase in α resulted in a large reduction in the total goodput of

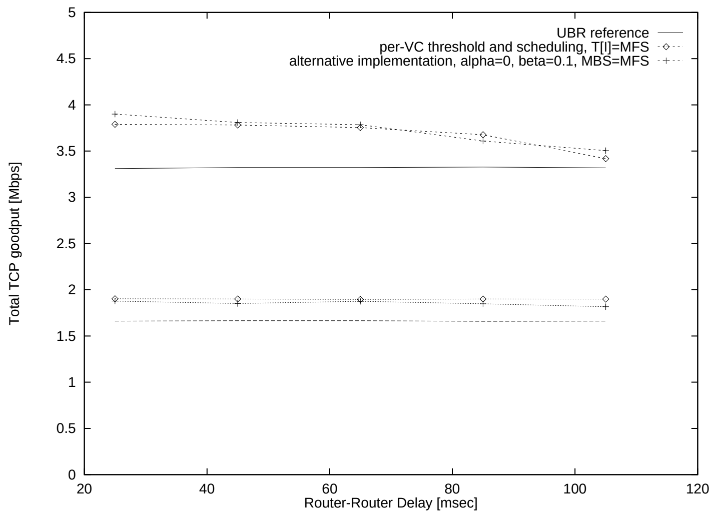

Figure 8.24: Ten routers scenario with TBA switch implementation

the workstations attached to the 4 Mbps router with a large delay. These results are not too surprising. When α is small, the packet discard probability is mainly a function of the value of the C[i] counter, and thus the probabilistic discards mainly affect the VCs which are mainly responsible for the congestion. On the other hand, when the value of α increases, the packet discard probability becomes more dependent on the average buffer occupancy than on the value of the C[i] counter.

α	β	Total goodput for 4 Mbps router with delay=105msec
0	0.05	3.2 Mbps
0	0.1	3.5 Mbps
0	0.2	3.5 Mbps
0	0.1	3.5 Mbps
0.0125	0.1	3.3 Mbps
0.05	0.1	3.1 Mbps

Table 8.1: Influence of α and β on total goodput with MBS=MFS

The simulations reported in this section show that, from a pure performance point of view, our TBA switch implementation could be a valid alternative to the per-VC threshold and scheduling implementation in ATM switches (e.g. backbone switches) which are used to carry internetwork traffic.

8.3.6 Related Work

Several implementations have been proposed in the literature to support the GFR service category in ATM switches. These implementations can be grouped in three main categories.

The simplest implementations do not rely on per-VC accounting but instead rely on the tagging of the non-eligible AAL5-PDUs performed by a F-GCRA at the ingress of the network. The main advantage of these implementations is their low complexity. This group of implementations include the Double-EPD [GH96] buffer acceptance algorithm and the modified Double-EPD buffer acceptance algorithm proposed in [EA98]. The main drawback of these algorithms is that they only support the GFR.2 conformance definition.

The implementations of the second group require per-VC accounting, but not per-VC queuing and scheduling. These implementations include our TBA switch implementation, Differential Fair Buffer Allocation (DFBA) [GJFV98], the Virtual Queueing technique proposed in [SWR97] and the second and third implementations proposed in [EA98] and discussed in section 7.3.7.

Finally, the implementations of the third group require both per-VC accounting and per-VC queuing and scheduling. An example from the third group is the per-VC threshold and scheduling implementation [GH96] described in section 7.2.2.

Weighted Fair Buffer Allocation

The Weighted Fair Buffer Allocation (WFBA) buffer acceptance algorithm [HK98] was proposed before the beginning of the work on GFR within the ATM Forum, but it could be used to support, at least partially, the GFR service category in ATM switches. WFBA relies on per-VC accounting and can be used in FIFO-based switches. It tries to allocate the space available inside a buffer to all the active VCs in proportion to a weight associated with each VC. For the GFR service category, this weight would be set to the MCR of the VC. When the buffer is full, WFBA tries to provide the same allocation as the static buffer reservation discussed in section 8.3.1. However, if the buffer occupancy is much smaller, then WFBA allows the active VCs to temporarily utilize a larger fraction of the buffer than the static buffer allocation. WFBA automatically adapts the fraction of the buffer which can be used by each VC in function of the weights of the active VCs and the occupancy of the buffer.

WFBA uses two buffer thresholds (K and R). K is similar to an EPD threshold. When the buffer occupancy (B_{Occ}) is below the R threshold, then the switch is assumed to be not congested and all the arriving AAL5-PDUs are accepted, independently of the buffer utilization of each VC. When the buffer occupancy is above this threshold, WFBA accepts or discards the arriving AAL5-PDUs as a function of the actual buffer occupancy of their respective VCs. For this, WFBA keeps track of the number of cells from each VC in the buffer ($B_{Occ}[i]$). In addition to the buffer size (K), the WFBA algorithm also uses a tuning parameter (Z, $0 < Z \leq 1$). When a new AAL5-PDU arrives on VC[i] and the total buffer occupancy is above the R threshold, it is accepted if equation 8.6 is true, where an active VC is a VC which has at least one cell inside the buffer.

$$B_{Occ}[i] \ \leq \ Z \times B_{Occ} \times \left(\frac{MCR[i]}{\sum_{j=1}^{Active\ VCs} MCR[j]}\right) \times \left(1 + \frac{K - B_{Occ}}{B_{Occ} - R}\right) \tag{8.6}$$

We have studied the suitability of WFBA to provide a minimum guaranteed bandwidth to internetwork traffic in [Bon98b]. In this paper, we used the "five routers" and the "ten routers" simulation scenarios. With the "five routers" scenario, WFBA achieved almost the same performance as the per-VC threshold and scheduling implementation. With the "ten routers" scenario, the round-trip-time had a higher impact with WFBA than with the per-VC threshold and scheduling implementation. However, as already mentioned in [Bon98b], WFBA is not sufficient to completely support the GFR service category in ATM switches since it cannot support the MBS part of the GFR traffic contract and has problems to support GFR VCs with a low or zero MCR.

Differential Fair Buffer Allocation

The DFBA implementation [GJFV98] aims at supporting the GFR service category in FIFO-based switches by using only per-VC accounting. DFBA utilizes some of the features of the Double-EPD, WFBA and RED buffer acceptance algorithms. DFBA handles the CLP=1 AAL5-PDUs like the Double-EPD implementation, i.e. they are discarded when the total occupancy of the FIFO buffer is above the LBO threshold. The CLP=0 AAL5-PDUs are always accepted when the occupancy of the FIFO buffer is below the LBO threshold and discarded when the occupancy of the FIFO buffer is above the HBO threshold. When the buffer occupancy is between the LBO and the HBO thresholds, the arriving CLP=0 AAL5-PDUs are discarded with a probability given by equation 8.7 where α is a global parameter set to 0.5 in [GJFV98], B_{Occ} is the total occupancy of the buffer, $B_{Occ}[i]$ is the number of cells from VC[i] in the buffer and $Z[i]$ is a per-VC parameter [12] .

$$P = Z[i] \times (\alpha \times \frac{B_{Occ}[i] - B_{Occ} \times \frac{MCR[i]}{\sum_j MCR[j]}}{B_{Occ} \times (1 - \frac{MCR[i]}{\sum_j MCR[j]})} + (1-\alpha) \times \frac{B_{Occ} - LBO}{HBO - LBO}) \tag{8.7}$$

[GJFV98] does not explicitly mention whether DFBA was designed to support the GFR.1 or the GFR.2 conformance definition. When used to support the GFR.2 conformance definition, DFBA would be similar to the Double-EPD implementation, although DFBA could discard CLP=0 AAL5-PDUs which would not be discarded by Double-EPD. This means that DFBA could fail to provide the minimum guaranteed bandwidth by discarding CLP=0 AAL5-PDUs. When considering the GFR.1 conformance definition, DFBA does not support the MBS part of the minimum guaranteed bandwidth.

Compared with DFBA, our TBA switch implementation provides several benefits. First, it allows the network operator to control how the unreserved bandwidth is distributed among the active VCs by using the administrative weights. This kind of control can be partially performed by DFBA thanks to the $Z[i]$ parameter. However, this parameter has only an indirect influence on the distribution of the unreserved bandwidth and its tuning seems to be rather difficult [GJFV98]. Second, our TBA switch implementation is able to support best-effort (i.e. UBR and GFR with MCR=0 VCs) in the same FIFO buffer as the GFR VCs with a non-zero MCR. This problem is not addressed in [GJFV98]. Third, our TBA switch implementation provides a complete support for the GFR.1 and GFR.2 conformance definitions, including the MBS part of the minimum guaranteed bandwidth for GFR.1. [GJFV98] is unclear concerning the supported GFR conformance definitions.

[12] [GJFV98] proposes to set $Z[i]$ equal to $1 - \frac{MCR[i]}{\sum_j MCR[j]}$ or $(1 - \frac{MCR[i]}{\sum_j MCR[j]})^2$.

The simulations discussed in [GJFV98] have studied the performance of the DFBA implementation with internetwork traffic. These simulations show that with DFBA the workstations attached to the routers are able to utilize their minimum guaranteed bandwidth. However, DFBA tends to allow the VCs with a small MCR to utilize a larger fraction of the unreserved bandwidth than the VCs with a large MCR, although this problem can be partially reduced by a careful selection of the $Z[i]$ parameter. [GJFV98] does not study the performance of DFBA with workstation traffic. Unfortunately, as [GJFV98] does not compare the performance of DFBA with other switch implementations such as Double-EPD or per-VC threshold and scheduling, it is impossible to compare quantitatively the performance of DFBA with our TBA switch implementation.

Virtual Queueing

Like our TBA switch implementation, the buffer acceptance algorithm proposed in [SWR97] aims at supporting the GFR service category with a single FIFO buffer. This implementation also relies on a buffer acceptance algorithm and a counter update algorithm, but these algorithms are completely different from the algorithms used in our implementation.

The buffer acceptance algorithm (figure 8.25) relies on one counter (M[i]) for each VC, uses one global threshold (Th) and keeps track of the number of active VCs. When the first cell of an AAL5-PDU arrives, it is accepted in the FIFO buffer if the total buffer occupancy is below the Th threshold or if the counter of this VC is below Th divided by the number of active VCs. In [SWR97], a VC is considered to be active when its M[i] counter is larger than $-w$, where w is a global parameter.This is similar to the definition of an active VC in our TBA switch implementation, but by using a global parameter, [SWR97] implicitly assumes that all VCs have the same MBS. When a cell is accepted in the FIFO buffer, the value of the M[i] counter is incremented by one. The buffer acceptance algorithm used by [SWR97] maintains a linked list "L" with the identifiers of all the active VCs. This list is used by the counter update algorithm.

The counter update algorithm is used to decrement the M[i] counters of the active VCs at a rate at least equal to the MCR of the corresponding VC. This algorithm (figure 8.26) uses one timer (T) and one counter (W[i]) for each VC. The counter update algorithm relies on the linked list "L" to identify the active VCs and operates in two phases during a period of T seconds. At the beginning of each period, the counter update algorithm operates in phase one. During this phase, the counters of the active VCs are decremented by at least MCR[i]$\times$T. This allows the counter update algorithm to provide the minimum guaranteed bandwidth. Phase 1 ends when the counters of all the active VCs have been sufficiently decremented. During the second phase of the algorithm, the M[i] counters of the active VCs are decremented in a round-robin fashion in order to equally distribute the unreserved (and unused reserved) bandwidth among the active VCs.

Compared with this implementation our TBA switch implementation provides the following benefits. First, our implementation distributes the unreserved bandwidth in proportion to the administrative weights of the active VCs, while the implementation discussed in [SWR97] only provides a fair allocation of the unreserved bandwidth. Second, we have shown how to support best-effort and nrt-VBR VCs in the same FIFO buffer as the GFR VCs. We have also explained how our proposed implementation could be used in a multi-service switch. These problems are not addressed in [SWR97]. Third, we have

```
Arrival of the first cell of an AAL5-PDU :
$Th_n$ = Th/$N_{active}$;
if ( ($B_{Occupancy}$ $\geq$ Th) and ( M[i] $\geq$ $Th_n$ ) )
{
  accept_PDU[i]=FALSE;
  discard_cell; /* Cell is discarded */
}
else
{
  /* Cell is accepted in the FIFO buffer */
  M[i]=M[i]+1;
  if ( M[i] = -$w$ +1)
    append VC[i] to the tail of linked list L;
  accept_PDU[i]=TRUE;
}

Arrival of a middle cell of an AAL5-PDU :
if (accept_PDU[i]=FALSE)
  discard_cell;
else
{
  /* Cell is accepted in the FIFO buffer */
  M[i]=M[i]+1;
  if ( M[i] = -$w$+1)
    append VC[i] to the tail of linked list L;
}
```

Figure 8.25: Buffer acceptance algorithm proposed in [SWR97]

shown how to maintain a low average buffer occupancy while still providing the minimum guaranteed bandwidth. This control of the buffer occupancy is an important feature for TCP/IP traffic which is not addressed in [SWR97]. A fourth benefit of our TBA switch implementation is that we have shown how to completely support the GFR.1 and GFR.2 conformance definitions. [SWR97] does not discuss the support of the two different GFR conformance definitions.

[SWR97] briefly studies the performance of the Virtual Queueing implementation by considering UDP and TCP-based workstation traffic in a LAN environment. In both cases, the simulations described in [SWR97] show that this implementation is able to provide the minimum guaranteed bandwidth. However, concerning the TCP simulations, it should be noted that [SWR97] considers a granularity of $10\mu sec$ for the retransmission timer. Such a small granularity is an unrealistic model of current software-based TCP implementations. The performance of the implementation proposed in [SWR97] would probably be worse with a realistic granularity for the retransmission. Unfortunately, this more realistic case is not studied in [SWR97]. [SWR97] does not analyze the performance of the Virtual Queueing implementation with internetwork traffic and does not compare

```
Timer expiration (period T) :

for(j=1;j≤N;j=j+1)
  W[j]=W[j]+MCR[j]*T;
phase=1;

When a cell from the FIFO buffer is transmitted :

if (phase=1)
{
  if (L is not empty)
  {
    let k denote the first VC identifier in L;
    remove k from L;
    M[k]=M[k]-1;
    W[k]=W[k]-1;
    if ( (M[k]>-w) and (W[k]>1))
      append k to the tail of L;
  }
  phase=2;
  for(j=1;j≤N; j=j+1)
    if (W[j] ≥ 1)
      phase=1;
}
else
{ /* phase2 */
  if( L is not empty)
  {
    let k denote the first VC identifier in L;
    remove k from L;
    M[k]=M[k]-1;
    if (M[k]>-w)
      append k to the tail of L;
  }
}
```

Figure 8.26: Counter update algorithm proposed in [SWR97]

its performance with other implementations such as Double-EPD and per-VC threshold and scheduling. It is thus impossible to compare quantitatively the performance of the implementation proposed in [SWR97] with our TBA switch implementation.

8.4 Improving per-VC threshold and scheduling

From our experience with the per-VC threshold and scheduling implementation described in chapter 7 and the simulations with our TBA switch implementation, we can propose two directions to improve the per-VC threshold and scheduling implementation. The first improvement concerns its buffer acceptance algorithm. Our simulations have shown that it had a strong bias in favor of GFR.1 VCs over GFR.2 VCs. As already mentioned, this bias is probably not acceptable in public networks. The buffer acceptance algorithm proposed in the section 8.3.3 for our TBA switch implementation could be used with small modifications in a switch based on the per-VC threshold and scheduling implementation. Instead of basing the acceptance decision on the value of the C[i] counter, it would be sufficient to base the acceptance on the number of cells in each per-VC queue. A possible solution would be to replace all occurrences of C[i] in figures 8.18 and 8.17 by $(MBS[i] - B_{Occ}[i])$ where $B_{Occ}[i]$ is the number of cells in the queue for VC[i]. The main characteristic of the buffer acceptance algorithm used in the TBA switch implementation, namely the complete support for GFR.1 and GFR.2 combined with a low average buffer occupancy thanks to the utilization of a variant of the RED algorithm, are also desirable in a switch with per-VC queueing and scheduling.

The second improvement concerns the scheduler used by the per-VC threshold and scheduling implementation. The per-VC threshold and scheduling implementation assumes the utilization of a WFQ-like scheduler. Many WFQ-like schedulers [Zha95] have been proposed since the original WFQ scheduler [DKS90]. The Virtual Spacing scheduler considered in this thesis is one of these WFQ-like schedulers. If we ignore implementation details, a common characteristic among these schedulers is that, when considering greedy sources, the bandwidth is shared in proportion to the weight associated with each VC. We have already shown that this distribution of the available bandwidth is not sufficient to efficiently support the GFR service category in ATM switches.

We show in this section how the Virtual Spacing algorithm can be extended to support a similar method as our TBA switch implementation to distribute the available bandwidth among the active VCs. We first recall some implementation issues when considering Virtual Spacing in section 8.4.1. We then propose in section 8.4.2 a modification to Virtual Spacing to support the same distribution of the available bandwidth as our TBA switch implementation. Finally we discuss two possible ways to reduce the implementation complexity of this scheduler in section 8.4.3.

8.4.1 Implementing Virtual Spacing

As explained in section 7.2.2, Virtual Spacing [Rob94] is usually described by assuming that one timestamp is associated with each arriving cell. However, this is not strictly necessary in practice. In fact, in a switch with per-VC queueing, only the timestamp of the first cell of each queue needs to be computed and stored [Rob94]. Thus, instead of requiring one timestamp per-cell, a real implementation of Virtual Spacing requires only one timestamp per queue.

We show below how Virtual Spacing can be implemented with one timestamp per queue. We assume an output buffered switch which supports N different VCs (numbered from 0 to N-1). We also assume that there is a one-to-one mapping from the VCs to the queues (i.e. Queue[k] contains only the cells from VC[k]) and we use the following notations :

- Queue[j].MCR = Minimum Cell Rate (MCR) for VC[j]
- Queue[j].F = Virtual finish time (timestamp) of the first cell of Queue[j]
- Queue[j].first = first cell of VC[j]

In addition to these per-queue variables, we also define one global variable :

- Vtime = Virtual time of the scheduler[13]

With these notations, the Virtual Spacing algorithm can be expressed as shown in figures 8.27 and 8.28. Figure 8.27 shows how the timestamp of a cell arriving in an empty queue is computed. Figure 8.28 describes the algorithm which selects the cell to be transmitted on the output link.

```
Cell arrival in Queue[j] :


if (Queue[j] is empty)
    Queue[j].F =Vtime + 1/Queue[j].MCR;
insert cell at tail of Queue[j];
```

Figure 8.27: Virtual Spacing (Cell arrival)

8.4.2 A flexible scheduler

The counter update algorithm based on the W_R and W_U registers which is part of our TBA switch implementation could also be used, with minor modifications, as a scheduler. Instead of selecting a counter to be updated this algorithm could select a cell to be transmitted. This algorithm could in fact be considered as a scheduler composed of two building blocks. The first building block, implemented by the W_R register, is equivalent to a non-work-conserving scheduler. The second building block, implemented by the W_U register, is equivalent to a work-conserving scheduler.

The Virtual Spacing algorithm is a work-conserving scheduler and thus it could be used to replace the W_U register. Virtual Spacing could also be used, with a small modification, to implement a non-work-conserving scheduler. For this, we note that with Virtual Spacing, the timestamp of the first cell of each VC (Queue[j].F) corresponds to the time at which this cell should be sent, at the latest, to fulfill the minimum guaranteed bandwidth of this VC. In a non-work-conserving scheduler, the different VCs should thus be served at the time corresponding to their timestamp.

The non-work-conserving version of Virtual Spacing is composed of two parts. The cell arrival part is exactly the same as figure 8.27. The cell transmission part (figure 8.29)

[13] In [Rob94], Vtime is called the "Spacing time".

```
Cell transmission :


/* assumes that there is at least one cell in one queue */

/* Find smallest F timestamp */
Fmin= $+\infty$;
s = -1;
for (k=0;k < N;k=k+1)
    if(Queue[k] not empty) AND (Queue[k].F $\leq$ Fmin)
    {
        Fmin=Queue[k].F;
        s=k;
    }
/* s points to the Queue with the smallest F */
tx = Queue[s].first;
remove_first(Queue[s]);
Vtime = Fmin; /* update spacing time */
if (Queue[s] is not empty)
    Queue[s].F = Vtime+1/Queue[s].MCR;
transmit_cell(tx);
```

Figure 8.28: Virtual Spacing (cell transmission)

is slightly different from the work-conserving scheduler. The main modification is that the non-work-conserving scheduler will only transmit a cell provided that its timestamp is larger than or equal to the real-time. The non-work-conserving scheduler must thus maintain a real-time clock. This real-time clock must be expressed in units which are compatible with the units used to define the MCR associated with each logical queue. For example, if the MCRs are expressed in cells per second, then the real-time clock should be incremented by one every cell slot (i.e. every $1/PCR_o$ seconds).

Based on the non-work-conserving version of Virtual Spacing, we can now describe our flexible scheduler. This scheduler is a two level scheduler. The provision of the minimum guaranteed bandwidth is performed by a non-work-conserving scheduler (the R-scheduler) and the bandwidth which is not used by this scheduler is distributed among the active VCs with a work-conserving scheduler (the U-scheduler). These two schedulers are based on Virtual Spacing, and we explain below how to combine them to provide a flexible scheduler. We associate the following variables with each queue:

- Queue[j].MCR : Minimum Cell Rate (MCR) associated with VC[j] (expressed in cells per second)

- Queue[j].W : Administrative weight associated with Queue[j] (unitless[14])

[14]The administrative weight of VC[j] is used to specify the unreserved and unused reserved bandwidth that VC[j] will be allowed to use.

```
Every cell slot :
/* assumes that there is at least one cell in one queue */

/* Find smallest F timestamp */
Fmin=+∞;
s = -1;
for (k=0;k<N;k=k+1)
{
  if(Queue[k] not empty) AND (Queue[k].F ≤ Fmin)
    {
      Fmin=Queue[k].F;
      s=k;
    }
}
/* s points to the Queue with the smallest F */
if (Fmin < time)
{
  tx = Queue[s].first;
  remove_first(Queue[s]);
  Vtime=Fmin;
  if (Queue[s] is not empty)
  {
    Queue[s].F = Fmin +1/Queue[s].MCR;
  }
  transmit_cell(tx);
}
else
{
  /* no cell is transmitted, server is idle */
}
```

Figure 8.29: Cell transmission for non-work-conserving Virtual Spacing

- Queue[j].F_R : Virtual finish time (timestamp) of the first cell of Queue[j] for the R-scheduler

- Queue[j].F_U : Virtual finish time (timestamp) of the first cell of Queue[j] for the U-scheduler

The F_R and F_U variables are used internally by the R- and U-schedulers. The administrative weight is used to determine how the unreserved (and unused reserved) bandwidth is distributed among the active VCs.

In addition to these per-queue variables, the flexible scheduler maintains the following three global variables :

- Vtime_U : Virtual time for the U-scheduler

- Vtime_R : Virtual time for the R-scheduler
- time : the real time clock

```
Cell arrives in Queue[j] :

if (Queue[j] is empty)
{
  Queue[j].F_U = Vtime_U + 1/Queue[j].W;
  Queue[j].F_R = Vtime_R + 1/Queue[j].MCR;
}
insert cell at tail of Queue[j];
```

Figure 8.30: Cell arrival for flexible scheduler

The flexible scheduler is shown in figures 8.30 and 8.31. Figure 8.30 describes the computation of the timestamps for the R- and U-schedulers when a cell arrives in an empty queue. This computation is done as in the classical Virtual Spacing algorithm.

Figure 8.31 describes the algorithm used by the flexible scheduler to select the cell to be transmitted on the output link. This algorithm is divided in two steps[15] . First, the R-scheduler determines whether one queue must be served to fulfill its minimum guaranteed bandwidth. For this, the R-scheduler compares the value of the smallest R timestamp with the real-time clock. If the smallest R timestamp is smaller than or equal to the real-time clock, then one cell must be transmitted on the output link. In this case, the first cell of the queue corresponding to the smallest R timestamp is transmitted on the output link and the R timestamp of this queue is updated provided that this queue is not empty. It should be noted that the U timestamp of this queue is not modified. If the smallest R timestamp is larger than the real-time clock, then all the active VCs have already received their minimum guaranteed bandwidth. In this case, the U-scheduler is invoked and the first cell from the queue with the smallest U timestamp is transmitted on the output link. After the transmission of this cell, the U timestamp (but not the R timestamp) of the corresponding is updated provided that this queue is not empty.

8.4.3 Efficient implementations of the flexible scheduler

When considering a hardware-based implementation of a scheduler, the main problem is usually that if N VCs are active, then the scheduler must find the smallest timestamp among N timestamps during each cell slot. Several solutions have been proposed recently [RGB96, BSZ97] to simplify the implementation of WFQ-like schedulers in ATM switches. These techniques are also applicable to the flexible scheduler described in the previous section.

[15]In figure 8.31, the F_R timestamps are first sorted and then the F_U timestamps are sorted if required. In a hardware-based implementation, we expect that these two sorting operations will be performed in parallel since they are independent.

```
Cell transmission :
/* assumes that there is at least one packet in one queue */
Fmin_R=+∞;
s_R = -1;
for (k=0;k<N;k=k+1)
  if(Queue[k] not empty) AND (Queue[k].F_R < Fmin_R)
  {
    Fmin_R=Queue[k].F_R;
    s_R=k;
  }
if (Fmin_R ≤ time)
{
  tx = Queue[s_R].first_cell;
  remove_first_cell(Queue[s_R]);
  Vtime_R=Fmin_R;
  if (Queue[s_R] is not empty)
    Queue[s_R].F_R = Fmin_R + 1/Queue[s_R].MCR;
  transmit_cell(tx);
}
else
{
  Fmin_U=+∞;
  s_U = -1;
  for (k=0;k<N;k=k+1)
    if(Queue[k] not empty) AND (Queue[k].F_U < Fmin_U)
    {
      Fmin_U=Queue[k].F_U;
      s_U=k;
    }
  tx = Queue[s_U].first_cell;
  remove_first_cell(Queue[s_U]);
  Vtime_U = Fmin_U;
  if (Queue[s_U] is not empty)
    Queue[s_U].F_U = Vtime_U + 1/Queue[s_U].W;
  transmit_cell(tx);
}
```

Figure 8.31: Cell transmission for flexible scheduler

The techniques proposed in [RGB96, BSZ97] reduce the complexity of the sorter by supporting only a small number of different weights (e.g. 64 in [BSZ97]). Such a solution can be applied to both the R- and the U-schedulers in our flexible scheduler. However, the most natural place to apply these techniques is the U-scheduler since this scheduler will usually only support a small number of different administrative weights. For simplicity, we only discuss the efficient implementation of the U-scheduler in this section.

In the remainder of this section, we consider two implementation techniques. The first technique is the one proposed in [BSZ97]. We will call this implementation the per-VC implementation because one timestamp is associated with each VC. This technique is applicable to the R- and the U-schedulers. For the second technique, we only use one timestamp per group of VCs, where all the VCs in one group have the same administrative weights. This technique is only applicable for the U-scheduler.

The per-VC implementation

The first implementation technique which can be used for the flexible scheduler is the solution proposed in [BSZ97]. This solution was originally proposed to support the WFQ_2^+ scheduler in ATM switches, but as mentioned in [BSZ97] it is also applicable for Virtual Spacing. This technique assumes that only R distinct weights are supported by the scheduler. In this case, only R timestamps are present in the sorter, even if there are much more than R active VCs.

A key contribution of [BSZ97] was to note that, with Virtual Spacing and a few other scheduling mechanisms used fixed length packets (e.g. ATM cells), when K VCs have the same weights, then it is possible to determine the smallest timestamp among these K VCs by maintaining a simple linked list and thus without any sorting operation. This interesting property of Virtual Spacing is due to several factors. The first factor is that the virtual time is, by definition, a non-decreasing function. The second factor is that, at any time, the virtual time is always smaller than or equal to the timestamps associated with all the active VCs. The third factor is that, when a cell arrives in an empty queue and after a cell has been scheduled for transmission, the timestamp associated with its VC is computed as $Vtime + 1/W$, where Vtime is the virtual time and W the weight associated with this VC.

Let us now consider a group of K VCs with the same weight. In this case, it is easy to show that the timestamps associated with the active VCs of this group are all bounded by :

$$\forall t, \forall i : 1 \leq i \leq K : Vtime(t) \leq TS[i](t) \leq Vtime(t) + 1/W \qquad (8.8)$$

In this equation, $Vtime(t)$ is the value of the virtual time of the scheduler at time t, $TS[i](t)$ is the timestamp associated with VC[i] at time t and W is the weight associated with the K VCs. The first part of this equation is true by definition of the virtual time. The second part of this equation can be easily proven by contradiction. Let us thus assume that at time t_1, $TS[j](t_1) > Vtime(t_1) + 1/W$. This timestamp was computed at some time $t_0 : t_0 < t_1$ according to the definition of the Virtual Spacing algorithm. At $t = t_0$, the following relation (8.9) was verified :

$$TS[j](t_0) = Vtime(t_0) + 1/W \qquad (8.9)$$

Furthermore, since the value of the timestamps only change when a cell is transmitted, we know that (8.10) :

$$\forall t : t_0 \leq t \leq t_1 TS[j](t) = TS[j](t_0) \qquad (8.10)$$

From the definition of the virtual time, the following property (8.11) is also true :

$$\forall t : t_0 \leq t, \quad Vtime(t) \geq Vtime(t_0) \tag{8.11}$$

When (8.9), (8.10) and (8.11) are combined, we have (8.12) at time $t = t_1$:

$$TS[j](t_1) \leq Vtime(t_1) + 1/W \tag{8.12}$$

(8.12) contradicts our initial hypothesis and thus the second part of (8.8) is true.

Based on this property, we can now explain how it is possible to find the smallest of the K timestamps without any sorting operation. For this, one linked list is associated with each group of K VCs. This list is either empty or contains identifiers of the active VCs of the group. By using appropriate insertion and deletions operations, it is possible to ensure that the VC identifier which is at the head of the list is always the VC with the smallest timestamp among all the active VCs of the group.

To understand the utilization of the linked list, let us assume that the list is initially sorted (this is trivial when the list contains only one VC identifier). When a cell arrives in an empty queue, the timestamp of its VC is computed as $TS[k] = Vtime + 1/W$. According to (8.8), this value is the upper bound of the timestamps associated with the K VCs. Thus, if the identifier of this VC is placed at the tail of the linked list, this linked list remains sorted. When a VC is selected for transmission, its VC identifier is removed from the linked list and the linked list remains sorted. The timestamp of this VC is then computed with the same formula as above and thus its VC identifier can also be safely placed at the tail of the linked list. Thus, the linked list remains sorted and at any time, the timestamp of the VC at the head of the linked list is the smallest timestamp among all the VCs of the group.

We show in figures 8.32 and 8.33 the modifications to the flexible scheduler to utilize this implementation technique. In these figures, a group is defined as the set of VCs with the same administrative weight. The following variables are used in these figures :

- Group[k].F_U: F_U timestamp associated with the group (i.e. smallest timestamp among all the active VCs of the group)
- Group[k].ll : Linked list associated with the group
- Group[k].ll.first : First element of the linked list (VC[i])
- N : number of VCs
- N_G : number of groups (i.e. number of different administrative weights)

In addition, we define two simple functions. GetUgroup(j) returns the identification of the group to which VC[j] belongs and Isempty(Group[k]) a boolean function which returns TRUE when the linked list associated with Group[k] is empty and FALSE otherwise.

Figures 8.32 and 8.33 described the cell arrival and the cell transmission part of the per-VC implementation of our flexible scheduler. When a cell arrives in an empty queue, its R and U timestamps are computed as in the flexible scheduler and the identifier of this queue is placed at the tail of the linked list of its group. If its group was empty, the timestamp of this group is set to the timestamp computed for the queue.

```
Cell arrives in Queue[j] :

if (Queue[j] is empty)
{
  Queue[j].F_U = Vtime_U + 1/Queue[j].W;
  Queue[j].F_R = Vtime_R + 1/Queue[j].MCR;
}
g=getUgroup(j);
if(Isempty(Group[g])
  Group[g].F_U=Queue[j].F_U;
insert j at tail of Group[g];
insert cell at tail of Queue[j];
```

Figure 8.32: Cell arrival for flexible scheduler with per-VC implementation

The algorithm used to select the cell to be transmitted on the output link (figure 8.33) works as follows. The R-scheduler is used as in the flexible scheduler to determine whether one queue must be served to fulfill its MCR. The only difference with the flexible scheduler described earlier is that we have to be careful when a queue containing a single cell is selected by the R-scheduler. When the first cell of this queue is transmitted, the queue becomes empty and thus its identifier must be removed from the linked list associated with its group. Otherwise, a timestamp corresponding to an empty queue would be included in the linked list.

The U-scheduler of the per-VC implementation works as follows. It first finds the smallest U-timestamp among the different groups. By construction, the queue associated with this timestamp is located at the head of the linked list associated with this group. The first cell from this queue is transmitted on the output link and the identifier of this queue is removed from the linked list. If the queue is not empty, its U-timestamp is computed as usual and the U-timestamp of the group is set to the U-timestamp of the queue which is located at the head of the linked list associated with the group.

```
Cell transmission :
/* assumes that there is at least one packet in one queue */
Fmin_R= $+\infty$;
s_R = -1;
for (k=0;k<N;k=k+1)
  if(Queue[k] not empty) AND (Queue[k].F_R < Fmin_R)
  {
    Fmin_R=Queue[k].F_R;
    s_R=k;
  }
if (Fmin_R $\leq$ time)
{
  tx = Queue[s_R].first_cell;
  remove_first_cell(Queue[s_R]);
  Vtime_R=Fmin_R;
  if (Queue[s_R] is not empty)
    Queue[s_R].F_R = Fmin_R + 1/Queue[s_R].MCR;
  else
  {
    g=getUgroup(s_R);
    remove s_R from list Group[g].ll;
  }
  transmit_cell(tx);
}
else
{
  Fmin_U=$+\infty$;
  s_U = -1;
  for (k=0;k< $N_G$;k=k+1)
    if(not(Isempty(Group[k])) AND (Group[k].F_U < Fmin_U))
    {
      Fmin_U=Group[k].F_U;
      s_U=k;
    }
  q = Group[s_U].ll.first;
  tx = Queue[q].first_cell;
  remove_first_cell(Queue[q]);
  Vtime_U = Fmin_U;
  remove q from list Group[s_U].ll;
  if (Queue[q] is not empty)
  {
    Queue[q].F_U = Vtime_U + 1/Queue[s_U].W;
    insert q at tail of list Group[s_U].ll;
  }
  if not(Isempty(Group[s_U]))
    Group[s_U].F_U=Queue[Group[s_U].ll.first].F_U;
  transmit_cell(tx);
}
```

Figure 8.33: Cell transmission for flexible scheduler with per-VC implementation

Per-group implementation

Although the implementation described in the previous section does not introduce any approximation when compared with the flexible scheduler, it might suffer, like the flexible scheduler, from one drawback. When considering greedy VCs, the flexible scheduler and the per-VC implementation distribute the available bandwidth according to equation 8.13, where U is the amount of unreserved (and unused reserved) bandwidth. This means that the fraction of the unreserved bandwidth which is allocated to one VC depends on the number of active VCs. This implies that a large number of VCs (e.g. best-effort) with a low administrative weight could utilize, as a whole, a large fraction of the unreserved bandwidth.

$$Rate[i] = MCR[i] + U \times \frac{W[i]}{\sum_j W[j]} \tag{8.13}$$

Some network operators could prefer another allocation of the unreserved bandwidth where the unreserved bandwidth is distributed among the active groups as shown in equation 8.14, where $N[k]$ is the number of VCs in the k^{th} group.

$$Rate[i] = MCR[i] + \frac{1}{N[getUgroup(i)]} \times (U \times \frac{Group[getUgroup(i)].W}{\sum_k Group[k].W}) \tag{8.14}$$

The advantage of this distribution of the available bandwidth is that the fraction of the unreserved bandwidth which is distributed to the VCs in one group does not anymore depend on the total number of VCs provided that there is always at least one active VC in each group, which seems to be a reasonable assumption.

This allocation of the available bandwidth can be easily performed by using the implementation shown in figures 8.34 and 8.35. In this implementation, a single timestamp is associated with each group of VCs. The successive values for these timestamps are computed by the classical Virtual Spacing algorithm. All the active VCs belonging to one group are served in a round-robin fashion every time their group is selected for transmission. This can be easily implemented with a simple linked list as shown in figure 8.35.

```
Cell arrives in Queue[j] :
if (Queue[j] is empty)
  Queue[j].F_R = Vtime_R + 1/Queue[j].MCR;
g=getUgroup(j);
if(Isempty(Group[g])
  Group[g].F_U=Vtime_U + 1/Group[g].F_U;
insert g at tail of Group[g];
insert cell at tail of Queue[j];
```

Figure 8.34: Cell arrival for flexible scheduler with per-group implementation

```
Cell transmission :
/* assumes that there is at least one packet in one queue */
Fmin_R=+∞;
s_R = -1;
for (k=0;k<N;k=k+1)
  if(Queue[k] not empty) AND (Queue[k].F_R < Fmin_R)
  {
    Fmin_R=Queue[k].F_R;
    s_R=k;
  }
if (Fmin_R ≤ time)
{
  tx = Queue[s_R].first_cell;
  remove_first_cell(Queue[s_R]);
  Vtime_R=Fmin_R;
  if (Queue[s_R] is not empty)
    Queue[s_R].F_R = Fmin_R + 1/Queue[s_R].MCR;
  else
  {
    g=getUgroup(s_R);
    remove s_R from list Group[g].ll;
  }
  transmit_cell(tx);
}
else
{
  Fmin_U=+∞;
  s_U = -1;
  for (k=0;k< $N_G$;k=k+1)
    if(not(Isempty(Group[k])) AND (Group[k].F_U < Fmin_U))
    {
      Fmin_U=Group[k].F_U;
      s_U=k;
    }
  q = Group[s_U].ll.first;
  tx = Queue[q].first_cell;
  remove_first_cell(Queue[q]);
  Vtime_U = Fmin_U;
  remove q from list Group[s_U].ll;
  if (Queue[q] is not empty)
    insert q at tail of list Group[s_U].ll;
  if not(Isempty(Group[s_U]))
    Group[s_U].F_U=Vtime_U+1/Group[s_U].W;
  transmit_cell(tx);
}
```

Figure 8.35: Cell transmission for flexible scheduler with per-group implementation

8.4.4 Related Work

Several two-level schedulers have been recently proposed in the literature. [CW97a] proposes to use a two-level scheduler to support ABR traffic in an ATM switches. In this article, a WRR-based scheduler is used as a non-work-conserving scheduler to provide the minimum guaranteed bandwidth to the ABR VCs with a non-zero MCR and a round-robin scheduler is used to distribute the unreserved bandwidth among all the active ABR VCs.

[SWR97] proposes another scheduler to support the GFR service category in ATM switches. This scheduler uses the algorithm shown in figure 8.26. It provides the minimum guaranteed bandwidth and distributes the unreserved bandwidth with a round-robin scheduler.

[BSZ97] uses a non-work-conserving $Staled-WFQ_2^+$ scheduler to perform traffic shaping, but also to provide minimum guaranteed bandwidth. In this case, the non-work-conserving scheduler is coupled with a round-robin scheduler to distribute the unreserved bandwidth equally among all the active VCs.

In addition to these schedulers, it is also useful to mention the link sharing algorithm proposed in [FJ95]. This algorithm was designed to share one link among different protocol families or organizations. This algorithm specifies the distribution of the bandwidth, but it does not take account of the flows with a minimum guaranteed bandwidth. A hierarchical scheduler also suitable for this purpose was proposed in [BZ97].

Compared with these proposals, our flexible scheduler provides an additional benefit to the network operator since he is not forced to distribute the unreserved bandwidth equally or in proportion of the minimum guaranteed bandwidth of the active VCs. We have also shown that when considering efficient implementations of our flexible scheduler it was possible to support two different strategies to distribute the unreserved bandwidth.

8.5 Conclusion

In this chapter, we have presented three important contributions to improve the performance of GFR switch implementations when carrying TCP/IP traffic.

We have first shown that the performance of workstation and internetwork traffic with the GFR.2 conformance definition could be significantly improved with some traffic shaping. We have proposed a simple shaper [Bon98c] which is suitable for the GFR service category and studied its performance. Although the simple shaper could probably be improved, our simulations with workstation and internetwork traffic have shown that the utilization of similar shapers significantly improves the performance of these two types of traffic, especially with the Double-EPD switch implementation.

Our second contribution is the TBA switch implementation which can be used to completely support the GFR service category in FIFO-based ATM switches. An important contribution of this switch implementation is that the distribution of the unreserved bandwidth is completely decoupled from provision of the minimum guaranteed bandwidth. Thanks to this decoupling, our TBA switch implementation allows the network operator to control the distribution of the unreserved bandwidth by choosing the administrative weights which are associated with each VC. This also allows our TBA switch implementation to support GFR VCs with a zero MCR without allocating an implicit minimum guaranteed bandwidth to these VCs. The simulations performed with our TBA switch implementation have shown that its performance was between the performance of the Double-EPD and per-VC threshold and scheduling implementations with workstation and internetwork traffic.

Our third contribution is a modification [BBD+98] to the Virtual Spacing scheduling algorithm. Based on our TBA switch implementation, we have shown how to modify Virtual Spacing in order to decouple the provision of the minimum guaranteed bandwidth from the distribution of the unreserved bandwidth among the active VCs. Our modification is important because it allows the network operator to directly control the distribution of the unreserved bandwidth to the GFR VCs with and without a minimum guaranteed bandwidth.

Chapter 9

Conclusion

In this thesis, we have presented a detailed study of the behaviour of TCP with ATM service categories providing a guaranteed minimum bandwidth. Our contributions can be classified into two groups. First, we have studied the suitability of the existing ATM service categories, namely Constant Bit Rate (CBR), Available Bit Rate (ABR) and Variable Bit Rate (VBR), to efficiently support TCP/IP service with a minimum guaranteed bandwidth. Our contributions related to these existing ATM service categories are summarized in section 9.1. Second, we have studied in detail the performance of TCP/IP with the Guaranteed Frame Rate (GFR) service category which is currently under study within the standardization bodies. We have also shown how to better support the GFR service category in ATM switches. Our contributions related to the GFR service category are summarized in section 9.2 Our work could be extended in several directions. We discuss these directions in section 9.3.

9.1 TCP and the existing service categories

Our first contribution is a detailed, measurement-based, performance evaluation of TCP with the CBR service category in a wide area ATM network [BKD96a]. The main conclusion of the measurements discussed in chapter 4 is that in such a network, it is very important to generate a conforming ATM traffic. If the ATM level traffic is not completely in conformance with the traffic contract, the Usage Parameter Control (UPC) function at the ingress of the ATM network discards cells, which causes a severe reduction in the performance of TCP. If the non-conformance of the ATM level traffic is too large, the data transfer becomes almost impossible with TCP.

We have studied the requirements to deploy the ABR service category in public network and shown that, with FIFO-based ABR switches, some kind of traffic enforcement was also needed to support ABR. We have shown that the utilization of such enforcement greatly increases the complexity of the private or public ATM switches depending on whether the public network relies on an ABR UPC or on switches with Virtual Source/Virtual Destination (VS/VD) capabilities. While ABR will probably be deployed in some corporate networks and inside public ATM networks to control the best-effort traffic at VP-level, we do not believe that ABR will be widely deployed in public networks.

As a simple alternative to the ABR service category, we have then studied in chapter 6 the performance of TCP with the VBR service category. In this chapter, we have first shown that, when considering workstation traffic in an ATM LAN relying on the VBR.3 conformance definition, TCP had huge difficulties to benefit from the minimum guaranteed bandwidth when all the cells tagged by the UPC were discarded inside the network [BKD96b]. We have studied the suitability of the VBR.3 conformance definition to support internetwork traffic. Our simulations have shown that internetwork traffic had difficulties to efficiently utilize VBR.3 VCs. We have also shown that in this case, the performance of TCP was highly dependent on the Maximum Burst Size (MBS) of the ATM VCs.

9.2 TCP and the GFR service category

Our study of TCP with the GFR service category discussed in chapters 7 and 8 has resulted in five important contributions.

Our first contribution is a detailed simulation study of the performance of the Double-EPD and the per-VC threshold and scheduling implementations with workstation traffic [Bon98d]. Our simulations have shown that the Double-EPD implementation is not a satisfactory solution to support TCP/IP traffic in LAN and WAN environments. We have shown that the low performance of the Double-EPD was mainly because the burstiness of the TCP traffic forced the Frame-Generic Cell Rate Algorithm (F-GCRA) used at the ingress of the network to tag a large fraction of the AAL5-PDUs sent by the workstations. Concerning the per-VC threshold and scheduling implementation, we have shown that it provides a much better performance than the Double-EPD implementation. However, we have shown that the performance of TCP becomes degraded in WAN environments and when the GFR.2 conformance definition is used.

Our second contribution is a detailed simulation study of the performance of these two switch implementations with internetwork traffic [Bon98b]. We have shown that internetwork traffic could better benefit from the minimum guaranteed bandwidth provided by the GFR service category than workstation traffic. Our simulations with Double-EPD implementation, have shown that the performance of internetwork traffic was very sensitive to the value of the MBS parameters of the GFR traffic contracts. With the Double-EPD implementation, large values for the MBS were required to achieve good performance. Our simulations with the per-VC threshold and scheduling implementation have shown that this implementation was well suited to support internetwork traffic when used with the GFR.1 conformance definition. However, we have identified two important drawbacks of this implementation. The first one is that it has difficulties to support GFR VCs with a zero Minimum Cell Rate (MCR) without implicitly reserving some bandwidth for these VCs. The second one is that when both GFR.1 and GFR.2 VCs are multiplexed in the same switch, the GFR.1 VCs are favored over the GFR.2 VCs which cannot benefit from their minimum guaranteed bandwidth.

Based on these simulations, we have then proposed three possible solutions to better support the GFR service category in ATM switches.

As a first solution to improve the performance of workstation and internetwork traffic with the GFR.2 conformance definition we have proposed the utilization of a simple shaper

[Bon98c]. We have shown that the utilization of this shaper significantly improved the performance of workstation and internetwork traffic with the Double-EPD implementation and that it reduced the unfairness between GFR.1 and GFR.2 VCs with the per-VC threshold and scheduling implementation. This shaper constitutes our third contribution.

We have then studied whether the GFR service category could be completely supported in FIFO-based ATM switches without requiring per-VC queueing. As a fourth contribution, we have proposed the Token-based Buffer Allocation algorithm (TBA) [Bon98a] which can be used in such switches. An important feature of our TBA switch implementation is that the distribution of the unreserved bandwidth is completely decoupled from the provision of the minimum guaranteed bandwidth. This decoupling allows our TBA switch implementation to support GFR VCs with a zero MCR without allocating an implicit minimum guaranteed bandwidth to these VCs. We have studied the performance of our TBA switch implementation with workstation and internetwork traffic. This simulation study has shown that the performance of our TBA switch implementation was close to the performance of the per-VC threshold and scheduling implementation with internetwork traffic. With workstation traffic, the performance of our TBA switch implementation was closer to Double-EPD than to the per-VC threshold and scheduling implementation. Although the performance of our TBA switch implementation lies between the Double-EPD and the per-VC threshold and scheduling implementation, its efficient support for best-effort VCs in addition to VCs with a minimum guaranteed bandwidth is an important improvement over the per-VC threshold and scheduling implementation.

Based on the work done to develop our TBA switch implementation, we have then shown how to modify a Virtual Spacing scheduler in order to decouple the provision of the minimum guaranteed bandwidth from the distribution of the unreserved bandwidth among the active VCs [BBD+98]. Our modification to the Virtual Spacing scheduler is a significant improvement since it allows this scheduler to support GFR VCs with a zero MCR without implicitly reserving some bandwidth for these VCs. Furthermore, our modification allows the network operator to directly control how the unreserved bandwidth is distributed among the active VCs. We have also discussed efficient implementation of our modified scheduler. This modified scheduler constitues our fifth contribution.

9.3 Further work

In this thesis, we have mainly relied on measurements and simulations to evaluate the performance of TCP over different ATM service categories.

We have used measurements to study the ATM service categories which are already supported by commercial products (i.e. CBR and VBR). During these measurements, we could only, due to the limited available equipment, utilize workstation traffic with only a few workstations. An interesting extension of these measurements would be to study the behaviour of the newer ATM service categories (e.g. ABR and GFR) in a network with at least several workstations. Such measurements will be performed during the upcoming months in the framework of an eight man year collaboration project that we have planned in late 1997. These measurements will only consider workstation traffic due to the cost of equipment, but they will study LAN, MAN and WAN environments thanks to the utilization of a delay emulator. Of course, the final test for the ATM service categories will be the deployment in real networks, but such a test is outside the capabilities of a research project.

Since the GFR service category is not yet supported by available equipment, we had to rely on simulations to evaluate the performance of TCP with this service category. Our simulations have proved to be very useful since they have identified several potential problems with the studied switch implementations. However, these simulations could be extended in several directions. The first one is to consider more complex networks than the simple networks we considered in this thesis. This research direction will be followed by the above mentioned research project. This project is composed of a measurement part which has been already described and a simulation part. The simulation is comparing the performance of TCP with several ATM service categories by considering a dozen of simulation scenarios including simple and complex networks. An important point of this simulation study is that all the simulations are carried out with the same simulator, the same simulation scenarios and the same traffic sources for all the ATM service categories. For the ATM service categories with a minimum guaranteed bandwidth, this research project considers the VBR, ABR, GFR and ATM Block Transfer (ABT) service categories. This research project also studies the performance of TCP with the main best-effort service categories (Unspecified Bit Rate (UBR), ABR and Controlled Transfer (CT)).

Another possible direction to improve the simulations described in this thesis would be to consider other types of TCP sources. In this thesis, we have only considered file transfers with relatively long files, which is a reasonable application for high speed ATM networks. It would be interesting to see how the different ATM service categories perform with applications such as web browsing. However, such simulations require an accurate model of the behaviour of a web browser and/or web server attached to a high-speed ATM network. Although there has been some recent work on the characterization of this application, it is not clear that the results obtained in a low speed Internet are also applicable for high-speed ATM networks. For example, the behaviour of a user attached to a 56 Kbps modem or a shared T1 link could be highly influenced by the speed of his connection to the Internet.

A third possible direction for further work based on this thesis would be to improve the flexible scheduler that we proposed in section 8.4. An interesting feature of this scheduler is that it efficiently supports VCs with and without a guaranteed minimum bandwidth on a single link. This feature is important to support the GFR service category, but is also applicable to other service categories such as ABR and VBR. A drawback of our proposed scheduler is that it maintains two timestamps for each VC and requires two sorters. Although we have shown how to reduce the complexity of our scheduler, it would be very useful, from a hardware implementation point of view, to maintain a single timestamp for each VC. A similar scheduler could also be used to efficiently support controlled-load [Wro97] together with best-effort flows in an IP router. However, modifying our flexible scheduler to support variable-length packets instead of fixed-length ATM cells would significantly increase its implementation complexity.

Bibliography

[AAA98] L. An, N. Ansari, and A. Arulambalam. TCP/IP traffic over ATM networks with FMMRA ABR flow and congestion control. *Computer Networks and ISDN Systems*, 29:2091–2102, 1998.

[ACMM97] H. Ahmed, R. Callon, A. Malis, and J. Moy. IP switching for scalable IP services. *Proceedings of the IEEE*, 85(12):1984–1997, December 1997.

[ADF$^+$98] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and R. Thomas. Label distribution protocol. Internet draft draft-mpls-ldp-00.txt, work in progress, March 1998.

[AHK$^+$97] S. Ano, T. Hasegawa, T. Kato, K. Narita, and K. Hokamura. Performance evaluation of TCP traffic over VBR in wide area ATM network. In A. Casaca, S. Tohme, J. Roberts, and J.P. Coudreuse, editors, *Proceedings of IEEE ATM97 Workshop*, pages 73–82, May 1997.

[AHK$^+$98] S. Ano, T. Hasegawa, T. Kato, K. Narita, and K. Hokamura. A study on accomodation of TCP/IP traffic using window scale option to international ATM network with VBR service category. In *1998 IEEE ICATM*, pages 484–491, June 1998.

[AKS96] R. Ahuja, S. Keshav, and H. Saran. Design, implementation and performance of a native mode ATM Transport layer. In *Proc. IEEE Infocom96*, 1996.

[All97] Quantum Flow Control Alliance. Quantum flow control specification, v2.0.5. available from http://www.qfc.org, March 1997.

[Alm92] P. Almquist. Type of service in the internet protocol suite. Internet RFC 1349, 1992.

[AT94] M. Ahmed and K. Tesink. Definitions of managed objects for ATM management version 8.0 using SMIv2. Internet RFC 1695, August 1994.

[ATe97] ATecoM GmbH. ATM_SHAP3, 32 class traffic shaper. Available from http://www.atecom.com/shap3.htm, 1997.

[Atk94] R. Atkinson. Default IP MTU for use over ATM AAL5. Internet RFC 1626, May 1994.

[BBB$^+$98] Y. Bernet, J. Binder, S. Blake, M. Carlson, E. Davies, B. Ohlman, D. Verma, Z. Wang, and W. Weiss. A framework for differentiated services. Internet draft draft-ietf-diffserv-framework-00.txt, work in progress, May 1998.

[BBD$^+$98] O. Bonaventure, P. Barri, E. Desmet, M. Henrion, and J. Verkinderen. A method to share available bandwidth, a processor realizing such a method, and a scheduler, an intelligent buffer and a telecommunication system including such a processor. European patent application number 98401959.6, July 31 1998.

[BCC$^+$98] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang. Recommendations on queue management and congestion avoidance. Internet RFC 2309, April 1998.

[BCS93] E. Biagioni, E. Cooper, and R. Sansom. Designing a practical ATM LAN. *IEEE Network Magazine*, 7(2):33–39, March 1993.

[BD98] O. Bonaventure and E. Desmet. Resolving the ambiguities of the GFR conformance definition. ATM Forum contribution 98-0281, April 1998.

[BDKD95] O. Bonaventure, A. Danthine, E. Klovning, and O. Danthine. TCP/IP and the European ATM Pilot. In M.Liu, editor, *1995 International Conference on Network Protocols (ICNP95)*, pages 270–277. IEEE Computer Society Press, November 1995.

[BF95] F. Bonomi and K. Fendick. The rate-based flow control framework for the Available Bit Rate ATM service. *IEEE Network Magazine*, 9(4):25–39, March-April 1995.

[BGSC92] P. Boyer, F. Guillemin, M. Servel, and J. Coudreuse. Spacing cells protects and enhances utilization of ATM network links. *IEEE Network Magazine*, 6(5):38–49, September 1992.

[BKD96a] O. Bonaventure, E. Klovning, and A. Danthine. Behaviour of TCP in the European ATM Pilot. *Computer Communication*, 19(3):264–275, March 1996.

[BKD96b] O. Bonaventure, E. Klovning, and A. Danthine. Is VBR a solution for an ATM LAN ? In W. Dabbous and C. Diot, editors, *Proc. Fifth IFIP Workshop on Protocols for High Speed Networks*, pages 60–74. Chapman and Hall, 1996.

[Bon97] O. Bonaventure. A simulation study of TCP with the proposed GFR service category. presented at Dagstuhl seminar on High Performance Networks for Multimedia Applications. Slides available from http://www.informatik.uni-mannheim.de/informatik/pi4/dagstuhl97/prog.html, technical report available from http://www-run.montefiore.ulg.ac.be/, June 1997.

[Bon98a] O. Bonaventure. A flexible buffer acceptance algorithm to support the GFR service category in ATM switches. In D. Kouvatsos, editor, *ATM'98 - sixth IFIP workshop on performance modelling and evaluation of ATM networks*, pages 71/1–71/10, July 1998.

[Bon98b] O. Bonaventure. Providing bandwidth guarantees to internetwork traffic in ATM networks. In *Proceedings of IEEE ATM98 workshop*, May 1998.

[Bon98c] O. Bonaventure. A shaping algorithm, a shaper realizing such a shaping algorithm and a communication network including such a shaper. European patent application number 98402245.9, September 12 1998.

[Bon98d] O. Bonaventure. A simulation study of TCP with the GFR service category. In A. Danthine, W. Effelsberg, D. Ferrari, and O. Spaniol, editors, *High Performance Networking for Multimedia Applications*. Kluwer Academic Publishers, 1998.

[BOP94] L. Brakmo, S. O'Malley, and L. Peterson. TCP Vegas: new techniques for congestion detection and avoidance. In *Proc. ACM SIGCOMM94*, pages 24–35, August 1994.

[Bra89] B. Braden. Requirements for Internet hosts - communication layers. Internet RFC 1122, October 1989.

[BSG92] P. Boyer, M. Servel, and F. Guillemin. The spacer-controller : An efficient UPC/NPC for ATM networks. In *Proceedings of the 14th International Switching Symposium*, Yokohama, Japan, 1992.

[BSZ97] J. Bennett, D. Stephens, and H. Zhang. High speed, scalable, and accurate implementation of fair queueing algorithms in ATM networks. In *Proc. ICNP'97*, 1997.

[BT92] P. Boyer and D. Tranchier. A reservation principle with application to the ATM traffic control. *Computer Networks and ISDN Systems*, 24:321–334, 1992.

[BZ97] J. Bennet and H. Zhang. Hierachical packet fair queueing algorithms. *IEEE/ACM Transactions on Networking*, October 1997.

[CHHC95] J. Crowcroft, A. Hilton, M. Handley, and S. Clayman. Experience with IP & ATM on the European PNO Pilot : shaping and policing. Message sent to the end2end-tf mailing list, March 1995.

[Cis98] Cisco. BXM-155 Stratm technology broadband switch module. available from http://www.cisco.com/warp/public/790/bpaxis/bxm_ds.htm, 1998.

[Cla82] D. D. Clark. Window and acknowledgement strategy in TCP. Internet RFC 813, July 1982.

[CMSB91] E. Cooper, O. Menzilcioglu, R. Sansom, and F. Bitz. Host interface design for ATM LANs. In *Proc. 16th Conference on Local Computer Networks*, October 1991.

[CT98] R. Coelho and S. Tohmé. A generic smoothing algorithm for real-time variable bit rate video traffic. *Computer Networks and ISDN Systems*, 29:2053–2066, 1998.

[CW97a] F. Chiussi and Y. Wang. An ABR rate-based congestion control algorithm for ATM. In *GLOBECOM97*, November 1997.

[CW97b] D. Clark and J. Wroclawski. An approach to service allocation in the Internet. Internet draft draft-clark-diff-svc-alloc-00.txt, work in progress, July 1997.

[CXK97] F. Chiussi, Y. Xia, and V. Kumar. Virtual techniques for ABR service : improving ABR/VBR interaction. In *INFOCOM97*, April 1997.

[Cyp96] R. Cyphers. AAL5 reassembly timeout. Fore Systems technical support, personal communication, 1996.

[Dav93] B. Davie. The architecture and implementation of a high-speed host interface. *IEEE Journal on Selected Areas in Communications*, 11(2):228–239, February 1993.

[DB94] O. Danthine and P. Boyer. Benefits of a spacer/controller in an ATM WAN : Preliminary traffic measurements. In *EXPLOIT Traffic Workshop*, Basel, Switzerland, 1994.

[DKS90] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. *Journal of Internetworking Research Practice and Experience*, 1:3–26, 1990.

[EA98] O. Elloumi and H. Afifi. Evaluation of FIFO-based buffer management for TCP over Guaranteed Frame Rate service. In *Proceedings IEEE ATM98 Workshop*, Fairfax, USA, May 1998.

[EARH95] O. Elloumi, H. Afifi, P. Rolin, and M. Hamdi. Issues in improving TCP performance over ATM. In *1st IFIP Conference on Traffic Management in ATM networks*, Paris, France, December 1995. Available from http://www.rennes.enst-bretagne.fr/~elloumi/index.html.

[Eng92] Protocol Engines. XTP protocol definition, revision 3.6. January 1992.

[FF96] K. Fall and S. Floyd. Simulation-based comparisons of Tahoe, Reno and SACK TCP. *ACM Computer Communication Review*, 26(3):5–21, July 1996.

[FJ93] S. Floyd and V. Jacobson. Random early detection for congestion avoidance. *IEEE/ACM Transactions On Networking*, 1(4):397–413, August 1993.

[FJ95] S. Floyd and V. Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM Transactions On Networking*, 3(4), August 1995.

[FKSS97] W. Feng, D. Kandlur, D. Saha, and K. Shin. On providing minimum rate guarantees over the Internet. IBM Research Report 20618, version 2, May 1997.

[FL97] C. Fang and A. Lin. TCP performance in ATM networks : ABR parameter tuning and ABR/UBR comparisons. In *Proceedings of SICON97*, 1997. Available from ftp://ftp-eng.cisco.com/users/alin/abr/sicon97-abr.ps.Z.

[For93] ATM Forum. *ATM User-Network Interface Specification V3.0.* Prentice-Hall, September 1993. ISBN 0-13-225863.

[For94a] ATM Forum. ATM User-Network Interface specification v3.1. ATM Forum specification af-uni-0010.002, September 1994.

[For94b] ATM Forum. Customer network management (CNM) for ATM public network service (M3 specification). ATM Forum specification af-nm-0019.000, October 1994.

[For95] ATM Forum. LAN Emulation over ATM 1.0. ATM Forum specification af-lane-0021.000, 1995.

[For96a] ATM Forum. ATM User Network Interface (UNI) Signaling specification, version 4.0. ATM Forum specification af-sig-0061.000, July 1996.

[For96b] ATM Forum. Private Network-Network Interface specification version 1.0 (PNNI 1.0). ATM Forum specification af-pnni-0055.000, March 1996.

[For96c] ATM Forum. Traffic Management 4.0. ATM Forum specification af-tm-0056.001, 1996.

[For97] ATM Forum. LAN Emulation over ATM version 2 - LUNI specification - straw ballot. ATM Forum specification af-str-lane-luni-02.00, April 1997.

[GAN91] R. Guerin, H. Ahmadi, and M. Naghshineh. Equivalent capacity and its application to bandwidth allocation in high speed networks. *IEEE Journal on Selected Areas in Communications*, 9(7):968–981, September 1991.

[Gar96] Mark Garrett. A service architecture for ATM : from applications to scheduling. *IEEE Network Magazine*, 10(3):6–14, May-June 1996.

[GCJ$^+$98] R. Goyal, X. Cai, R. Jain, S. Fahmy, and B. Vandalore. Per-VC rate allocation techniques for ATM-ABR virtual source virtual destination networks. submitted to GLOBECOM98, 1998.

[GH96] R. Guerin and J. Heinanen. UBR+ Service Category Definition. ATM Forum contribution ATM96-1598, December 1996.

[GH97] R. Guerin and J. Heinanen. UBR+ enhancements. ATM Forum contribution 97-0015, February 1997.

[GJF+97] R. Goyal, R. Jain, S. Fahmy, B. Vandalorre, S. Kalyanaraman, S. Kota, and P. Samudra. GFR – providing rate guarantees with FIFO buffer to TCP traffic. ATM Forum contribution 97-0831, September 1997.

[GJFV98] R. Goyal, R. Jain, S. Fahmy, and B. Vandalore. Buffer management for the gfr service. ATM Forum contribution 98-0405, July 1998.

[Gol94] S. Golestani. A self-clocked fair queuing scheme for broadband applications. In *IEEE INFOCOM94*, pages 636–646, 1994.

[Gui97] F. Guillemin. ATM Block Transfer capability versus Available Bit Rate service. *European Transactions on Telecommunications (ETT)*, 8(1):21–32, January-February 1997.

[Hei93] J. Heinanen. Multiprotocol encapsulation over ATM Adaptation Layer 5. Internet RFC 1483, July 1993.

[HK98] J. Heinanen and K. Kilkki. A fair buffer allocation scheme. *Computer Communications*, 21:220–226, 1998.

[HLG97] J. Heinanen, A. Lin, and R. Guérin. Removing network based tagging from GFR. ATM Forum contribution ATM97-0479, July 1997.

[HLK97] J. Huang, B. Lee, and S. Khorsandi. Simulation study of GFR implementations. ATM Forum contribution 98-1035, December 1997.

[Hoe96] J. Hoe. Improving the start-up behaviour of a congestion control scheme for TCP. In *Proc. ACM SIGCOMM96*, pages 270–280, August 1996.

[HOMM97] G. Hasegama, H. Ohsaki, M. Murata, and H. Myahara. Performance improvements of TCP over EFCI-based ABR service class by tuning congestion control parameters. *IECE Transactions on Communications*, E80-B(10):1444–1453, October 1997.

[HRR97] M. Hamdi, J. Roberts, and P. Rolin. Rate control for VBR video coders in broadband networks. *IEEE Journal on Selected Areas in Communications*, 15(6):1040–1051, August 1997.

[HS98] D. Hong and T. Suda. Performance of ERICA and QFC for transporting bursty TCP sources with bursty interfering traffic. In *Proceedings INFOCOM98*, April 1998.

[HV98a] F. Hellstrand and A. Veres. Simulation of TCP/IP router traffic over ATM using GFR and VBR.3. ATM Forum contribution 98-0087, February 1998.

[HV98b] E. Hernandez-Valencia. Refinements to service description and conformance definitions for GFR. Delayed contribution to ITU-T SG13 meeting, Question 7/13, June 1998.

[IKK97] M. Ishizuka, A. Koike, and M. Kawarasaki. Performance analysis of TCP over ABR in high-speed WAN environment. *IECE Transactions on Communications*, E80-B(10):1436–1443, October 1997.

[Inc96] FORE Systems Inc. Programmer's reference manual for AALI interface. Part #MANU0023. Available from ftp://ftp.fore.com, March 1996.

[Ins96] Texas Instruments. TNETA1585 programmer's reference guide. Available from http://www-s.ti.com/sc/psheets/sdnu016a/sdnu016a.pdf, December 1996.

[ISO84] ISO. Open Systems Interconnection - basic reference model. ISO 7498, 1984.

[ITU84] ITU-T. ISDN protocol reference model. ITU-T Recommendation I.320, 1984.

[ITU90] ITU-T. Broadband aspects of ISDN. ITU-T Recommendation I.121, 1990.

[ITU91] ITU-T. B-ISDN ATM Adaptation Layer (AAL) specification. ITU-T Recommendation I.363, 1991.

[ITU93a] ITU-T. B-ISDN ATM layer specification. ITU-T Recommendation I.361, 1993.

[ITU93b] ITU-T. Digital subscriber signalling system no. 1 (DSS 1) - ISDN user-network interface layer 3 specification for basic call control. ITU-T Recommendation Q.931, March 1993.

[ITU93c] ITU-T. Traffic control and congestion control in B-ISDN. ITU-T Recommendation I.371, 1993.

[ITU94a] ITU-T. B-ISDN ATM adaptation layer - service specific connection oriented protocol (SSCOP). ITU-T Recommendation Q.2110, July 1994.

[ITU94b] ITU-T. B-ISDN ATM signalling ATM adaptation layer - service specific coordination function for support of signalling at the user-network interface (SSCF at UNI). ITU-T Recommendation Q.2130, July 1994.

[ITU96a] ITU-T. B-ISDN ATM Adaptation Layer specification : Type 5 AAL. ITU-T Recommendation I.363.5, August 1996.

[ITU96b] ITU-T. B-ISDN ATM layer cell transfer performance. 05/96 draft 5R Rev ITU-T recommendation I.356, May 1996.

[ITU96c] ITU-T. Traffic control and congestion control in B-ISDN. ITU-T Recommendation I.371, August 1996.

[ITU97] ITU-T. Traffic control and congestion control in B-ISDN : conformance definition for ABT and ABR. ITU-T Recommendation I.371.1, 1997.

[Jac88] V. Jacobson. Congestion avoidance and control. In *Proc. ACM SIGCOMM88*, pages 314–329, August 1988.

[JB88] V. Jacobson and R. Braden. TCP extensions for long-delay paths. Internet RFC 1072, October 1988.

[JBB92] V. Jacobson, B. Braden, and D. Borman. TCP extensions for high-performance. Internet RFC 1323, May 1992.

[JBLC96] L. Jaussi, J.-M. Barcelo, S. Louis, and O. Casals. Experimental evaluation of CDV impact on ATM resource management. *European Transactions on Telecommunications (ETT)*, 7(5):407–421, September-October 1996.

[JKGF96] R. Jain, S. Kalyanaraman, R. Goyal, and S. Fahmy. Source behaviour for ATM ABR traffic management : and explanation. *IEEE Communications Magazine*, 34(11), November 1996.

[JY97] S. Jagannath and N. Yin. End-to-end traffic management in IP/ATM internetworks. In A. Casaca, S. Tohme, J. Roberts, and J.P. Coudreuse, editors, *Proceedings of the IEEE ATM'97 workshop*, pages 83–89, May 1997.

[KB96] E. Klovning and O. Bonaventure. Behaviour of XTPX in the European ATM Pilot. *European Transactions on Telecommunications (ETT)*, 7(5):444–454, September-October 1996.

[KDW$^+$98] H. Kim, T. Dwight, D. Whipple, W. Edwards, T. So, H. Suzuki, R. Klessig, and J. Heinanen. UBRw : Weighted UBR service. ATM Forum contribution 98-0445, July 1998.

[Ken98] J. Kenney. Traffic management baseline text document. ATM Forum document BTD-TM-01.01, April 1998.

[KHBG91] H. Kroner, G. Hébuterne, P. Boyer, and A. Gravey. Priority management in ATM switching nodes. *IEEE Journal on Selected Areas in Communications*, 9(3):418–427, April 1991.

[KJF$^+$97] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and B. Vandalore. The ERICA switch algorithm for ABR traffic management in ATM networks. Submitted to IEEE/ACM Transactions on Networking, available from http://www.cis.ohio-state.edu/˜jain, November 1997.

[KJF$^+$98] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and S. Kim. Performance and buffering requirements for internet protocols over ATM ABR and UBR services. *IEEE Communications magazine*, June, 1998.

[KJJ$^+$98] S. Kalayanaraman, R. Jain, J. Jiang, R. Goyal, S. Fahmy, and P. Smudra. Design considerations for the virtual source/virtual destination (VS/VD) feature in ABR service of ATM networks. To appear in Computer Networks and ISDN systems, 1998.

[KK95] E. Klovning and O. Kure. Host adapter spacing in SBA-200. Telenor Research (scientific document N17/95), 1995.

[KM95] H. Kung and R. Morris. Credit-based flow control for ATM networks. *IEEE Network Magazine*, 9(2):40–48, March/April 1995.

[KNME97] Y. Katsube, K. Nagami, S. Matsuzana, and H. Esaki. Internetworking based on cell switch router - architecture and protocol overview. *Proceedings of the IEEE*, 85(12):1998–2006, December 1997.

[KP87] P. Karn and C. Partridge. Improving round-trip time estimates in reliable transport protocols. In *Proc. ACM SIGCOMM87*, August 1987.

[KSC91] M. Katevenis, S. Srdiropoulos, and C. Courcoubetis. Weighted round-robin cell multiplexing in a general purpose ATM switch chip. *IEEE Journal on Selected Areas in Communications*, 9(8):1256–1279, 1991.

[LAH97] L. Lamti, H. Afifi, and M. Hamdi. Design and implementation of a flexible traffic controller for ATM connections. In A. Tantawi, editor, *High Performance Networking (HPN97)*, pages 130–146. Chapman and Hall, May 1997.

[Lau94] M. Laubach. Classical IP and ARP over ATM. Internet RFC 1577, January 1994.

[LBC97] S. Van Luinen, Z. Budrikis, and A. Cantoni. The Controlled Cell Transfer capability. *ACM Computer Communication Review*, 27(1):55–71, January 1997.

[Lee98] B. Lee. GFR dimensionning for TCP traffic. ATM Forum contribution 98-0271, April 1998.

[Lin96] A. Lin. Lightstream 1010 switch architecture and traffic management. Cisco Systems White paper. Available from http://www.cisco.com, 1996.

[LK98] D. Lin and H. Kung. TCP fast recovery strategies : analysis and improvements. In *Proc. INFOCOM98*, April 1998.

[LKP$^+$98] J. Luciani, D. Katz, D. Piscitello, B. Cole, and N. Doraswamy. NBMA next hop resolution protocol (NHRP). Internet RFC2332, April 1998.

[LMKQ89] S. Leffler, M. McKusick, M. Karels, and J. Quaterman. *The design and implementation of the 4.3BSD UNIX operating system.* Addison-Wesley, 1989.

[LNO96] T. Lakshman, A. Neidhardt, and T. Ott. The drop front strategy in TCP and in TCP over ATM. In *Proceedings INFOCOM96*, pages 1242–1250, 1996.

[LNST97] F. Ly, M. Noto, A. Smith, and K. Tesink. Definitions of supplemental managed objects for ATM management. Internet draft draft-ietf-atommib-atm2-09.txt, work in progress, February 1997.

[LP96] J.-R. Louvion and B. Piller. Performance measurements and traffic characterization in the European ATM Pilot. *European Transactions on Telecommunications (ETT)*, 7(5):455–466, September-October 1996.

[LST$^+$96a] H. Li, K. Siu, H. Tzeng, C. Ikeda, and H. Suzuki. On TCP performance in ATM networks with per-VC early packet discard mechanisms. *Computer Communications*, 19, 1065-1076 1996.

[LST$^+$96b] H. Li, K. Siu, H. Tzeng, C. Ikeda, and H. Suzuki. A simulation study of TCP performance in ATM networks with ABR and UBR services. In *Proceedings of INFOCOM96*, pages 1269–1276, 1996.

[Lyo91] Tom Lyon. Simple and efficient adaptation layer (SEAL). T1S1 document T1S1.5/91-292, August 1991.

[Mah96] J. Mahdavi. Experimental TCP selective acknowledgment implementation. Available from http://www.psc.edu/networking/tcp.html, 1996.

[Man96] S. Manthorpe. STCP 3.0 User Manual. Technical report, EPFL, September 1996.

[MB97] S. Manthorpe and J.Y. Le Boudec. A comparison of ABR and UBR to support TCP traffic. Technical Report DI 97-224, Ecole Polytechnique Fédérale de Lausanne, April 1997. Available from http://lrcwww.epfl.ch.

[MBCM95a] M. Ajmone Marsan, A. Bianco, R. Lo Cigno, and M. Munafo. Shaping TCP traffic in ATM networks. In *IEEE ICT '95*, Bali, Indonesia, April 1995. Available from http://hp0tlc.polito.it/~bianco/import/Papers_ps/bali.ps.

[MBCM95b] M. Ajmone Marsan, A. Bianco, R. Lo Cigno, and M. Munafo. Some simulation results about shaped TCP connections in ATM networks. In *3rd IFIP Workshop on Performance Modeling and Evaluation of ATM Networks*, Bradford, UK, July 1995. Available from http://hp0tlc.polito.it/~bianco/import/Papers_ps/bradford.ps.

[MD90] J. Mogul and S. Deering. Path MTU discovery. Internet RFC 1191, April 1990.

[Mee97] G. Meempat. Throughput behavior of reliable stream protocols in admission controlled bearer networks. *Computer Networks and ISDN Systems*, 29:165–179, 1997.

[MG95] K. Moldeklev and P. Gunningberg. How a large ATM MTU causes deadlocks in TCP data transfers. *IEEE/ACM Transactions On Networking*, 3(4):409–422, August 1995.

[Mic96] Sun Microsystems. ATM 622 adapter. Available from http://www.sun.com/, 1996.

[Mis96] P. Mishra. Effect of leaky bucket policing on TCP over ATM performance. In *Proceedings of ICC96*, June 1996.

[MKK94] K. Moldeklev, E. Klovning, and O. Kure. TCP/IP behaviour in a high-speed local ATM environment. In *Proc. 19th Conference on Local Computer Networks*, pages 176–185, Minneapolis, USA, 1994.

[MM96] J. Mahdavi M. Mathis. Forward acknowledgement: refining TCP congestion control. In *Proc. ACM SIGCOMM96*, pages 281–292, August 1996.

[MMFR96] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgment options. Internet RFC 2018, October 1996.

[MMR93] B. Metzler, I. Miloucheva, and K. Rebensburg. Specification of the broadband transport protocol XTPX. RACE 2060 CIO Deliverable 30, 1993.

[Mot97] Motorola. MPC860SAR communication controller technical summary. Available from http://www.mot.com/SPS/RISC/netcomm-/docs/pubs/MPC860SAR_TS.pdf, May 1997.

[Nag84] J. Nagle. Congestion control in TCP/IP internetworks. Internet RFC 896, January 1984.

[NBM+97] D. Niehaus, A. Battou, A. McFarland, B. Decina, H. Dardy, V. Sirkay, and B. Edwards. Performance benchmarking of signaling in ATM networks. *IEEE Communications Magazine*, 35(8):134–143, August 1997.

[NEC95] NEC. μpd98401 local ATM SAR chip, user's manual. Document No. IEU-1384A, August 1995.

[NEH+98] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Liaw, T. Lyon, and G. Minshall. Ipsilon's general switch management protocol specificiation version 2.0. Internet RFC 2297, March 1998.

[NH98] J. Nelissen and A. Hendrikse. Experimental TCP performance evaluation in an ABR access environment. In *Proc. GLOBECOM98*, 1998.

[NLM96] P. Newman, T. Lyon, and G. Minshall. Flow labelled IP : a connectionless approach to ATM. In *IEEE INFOCOM'96*, pages 1251–1260, March 1996.

[NMLH97] P. Newman, G. Minshall, T. Lyon, and L. Huston. IP switching and gigabit routers. *IEEE Communications magazine*, January 1997.

[NS97] P. Narvaez and K. Siu. An acknowledgement bucket scheme for regulating TCP flow over ATM. In *Proceedings of GLOBECOM97*, pages 1838–1844, October 1997.

[OA97] T. Ott and N. Aggarwal. TCP over ATM : ABR or UBR ? In *Proceedings of ACM SIGMETRICS97*, pages 52–63, 1997.

[Onv95] R. Onvural. *Asynchronous Transfer Mode Networks : Performance Issues.* Artech House, second edition, 1995.

[Ous90] J. Ousterhout. Why aren't operating systems getting faster as fast as hardware ? In *Proc. Summer 1990 USENIX Conference*, 1990. Similar measurements with recent workstations have been performed by H. Schulzrinne and may be found at http://www.fokus.gmd.de:80/step/hgs/bench/summary.html.

[PB97] S. Pappu and D. Basak. TCP over GFR implementation with different service disciplines : a simulation study. ATM Forum contribution atm97-0310, April 1997.

[PMH+95] M. Perez, F. A. Mankin, E. Hoffman, G. Grossman, and A. Malis. ATM signaling support for IP over ATM. Internet RFC 1755, February 1995.

[Pos81a] J. Postel. Internet protocol, protocol specification. Internet RFC 791, September 1981.

[Pos81b] J. Postel. Transmission control protocol, protocol specification. Internet RFC 793, September 1981.

[PR95] M. Perloff and K. Reiss. Improvements to TCP performance in high-speed ATM networks. *Communications of the ACM*, 38(2):90–100, February 1995.

[PRW+95] M. Parker, P. Robinson, R. Wade, D. Alley, P. Adam, B. Le Moine-Py, and P.B. Saint Hilaire. The European ATM pilot. In *Proceedings of the 15th International Switching Symposium (ISS'95)*, volume 1, pages 146–150, Berlin, April 1995.

[Pry93] M. De Prycker. *Asynchronous Transfer Mode : Solution for Broadband ISDN.* Ellis Horwood Series in Computer Communications and Networking. Ellis Horwood, second edition, 1993.

[PS97] Inc. PMC-Sierra. PMC-Sierra PM7375 LASAR-155. Available from http://www.pmc-sierra.bc.ca/atm/lasar.html, 1997.

[RAGG98] S. Rosenberg, M. Aissaoui, K. Gahway, and N. Giroux. Functionality at the edge : designing scalable multiservice ATM networks. *IEEE Communications magazine*, 36(5):88–99, May 1998.

[RBL98] V. Rosolen, O. Bonaventure, and G. Leduc. Impact of cell discard strategies on TCP/IP in ATM UBR networks. In D. Kouvatsos, editor, *ATM'98 - sixth IFIP workshop on performance modelling and evaluation of ATM networks*, pages 82/1–82/10, July 1998.

[RDR$^+$97] Y. Rekhter, B. Davie, E. Rosen, G. Swallow, D. Farinacci, and D. Katz. Tag switching architecture overview. *Proceedings of the IEEE*, 85(12):1973–1983, December 1997.

[RF94] A. Romanow and S. Floyd. The dynamics of TCP traffic over ATM networks. In *Proc. ACM SIGCOMM94*, pages 79–88, London, UK, September 1994.

[RF95] A. Romanow and S. Floyd. Dynamics of TCP traffic over ATM networks. *IEEE Journal on Selected Areas in Communications*, 13(4):633–641, May 1995.

[RGB96] J. Rexford, A. Greenberg, and F. Bonomi. Hardware-efficient fair queueing architectures for high-speed networks. In *Proc. INFOCOM96*, pages 638–646, 1996.

[Rob94] J. Roberts. Virtual spacing for flexible traffic control. *International Journal of Communication Systems*, 7:307–318, 1994.

[Rob97] L. Roberts. Explicit rate flow control : a 100 fold improvement over TCP. ATM Forum contribution 97-0462, 1997.

[SD97] C. Song and T. Dwight. Effect of packet-based policing on Internet traffic. ATM Forum contribution ATM97-0173, 1997.

[SDK98] R. Satyavolu, K. Duvedi, and S. Kalayanaraman. Explicit rate control of TCP applications. ATM Forum contribution 98-152R1, February 1998.

[SG198] ITU-T SG13. I.371 and I.371.1 living list. Output of the Geneva meeting, June 1998.

[SM84] T. Slattery and M. Muss. ttcp.c. available as ftp://ftp.brl.mil/pub/ttcp.c or ftp://www-run.montefiore.ulg.ac.be/pub/soft/ttcp2.c, 1984.

[SST95] J. Sterbenz, H. Schulzrinne, and J. Touch. Report and discussion on the IEEE ComSoc TCGN gigabit networking workshop 1995. *IEEE Network Magazine*, pages 9–21, July-August 1995.

[SSZ98] I. Stoica, S. Shenker, and H. Zhang. Core-stateless fair queueing: achieving approximately fair bandwidth allocation in high speed networks. In *Proc. ACM SIGCOMM98*, September 1998.

[ST96] K. Siu and H. Tzeng. Performance of TCP over ATM with time-varying available bandwidth. *Computer Communications*, 19:927–936, 1996.

[Ste94] W. Stevens. *TCP/IP Illustrated, volume 1 : The protocols.* Addison-Wesley, 1994.

[Ste97] W. Stevens. TCP slow start, congestion avoidance, fast retransmit and fast recovery algorithms. Internet RFC 2001, January 1997.

[SWR97] K. Siu, Y. Wu, and W. Ren. Virtual queueing techniques for UBR+ service in ATM with fair access and minimum bandwidth guarantee. In *GLOBE-COM'97*, pages 1081–1085, 1997.

[Sys95] FORE Systems. ForeRunner ASX-200BXE ATM switch user's manual. software version 3.3.x, July 1995.

[TS93] C. Traw and J. Smith. Hardware/software organization of a high performance ATM host interface. *IEEE Journal on Selected Areas in Communications*, 11(2):228–239, February 1993.

[VKJ+97] B. Vandalore, S. Kalyanaraman, R. Jain, R. Goyal, S. Fahmy, and S. Kim. Performance of bursty world wide web (www) sources over ABR. In *Proc. WebNet97*, November 1997.

[VKJ+98] B. Vandalore, S. Kalyanaraman, R. Jain, R. Goyal, S. Fahmy, and S. Kim. Simulation study of world wide web traffic over the ATM ABR service. submitted to SPIE98, 1998.

[VR98] P. Vaananen and R. Ravikanth. Framework for traffic management in MPLS networks. Internet draft draft-vaananen-mpls-tm-framework-00.txt, work in progress, March 1998.

[WC91] Z. Wang and J. Crowcroft. A new congestion control scheme : Slow start and search (tri-S). *ACM Computer Communication Review*, 21(1):32–43, January 1991.

[WC92] Z. Wang and J. Crowcroft. Eliminating periodic packet losses in 4.3-Tahoe BSD TCP congestion control. *ACM Computer Communication Review*, 22(2):9–16, April 1992.

[WC97] P. White and J. Crowcroft. Integrated services in the Internet : the next stage in Internet : state of the art. *Proceedings of the IEEE*, 85(12):1934–1946, December 1997.

[WC98a] P. White and J. Crowcroft. ATM switching and IP routing integration : the next stage in Internet evolution ? *IEEE Communications magazine*, April 1998.

[WC98b] D. Wu and H.J. Chao. TCP/IP over ATM-GFR. In *Proceedings IEEE ATM98 Workshop*, Fairfax, USA, May 1998.

[WNH98] T. Worster, B. Northcote, and J. Heinanen. GFR harmonization. ATM Forum contribution 98-0708, October 1998.

[WPM97] J. Witters, G. Petit, and E. Metz. Throughput analysis of a DGCRA-based UPC function monitoring misbehaving ABR end-systems. In A. Casaca, S. Tohme, J. Roberts, and J.P. Coudreuse, editors, *Proc. IEEE ATM97 workshop*, May 1997.

[Wro97] J. Wroclawski. Specification of the Controlled-Load network element service. Internet RFC 2211, September 1997.

[WS95] G. Wright and R. Stevens. *TCP/IP Illustrated Vol. 2, The Implementation.* Addison-Wesley, 1995.

[WSR97] Y. Wu, K. Siu, and W. Ren. Improved virtual queueing and dynamic EPD techniques for TCP over ATM. In *Proceedings ICNP97*, 1997.

[WW92] E. Wallmeier and T. Worster. The spacing-policer, an algorithm for efficient peak bit rate control in ATM networks. In *1992 International Switching Symposium (ISS92)*, volume 22-26, 1992.

[Zha86] L. Zhang. Why TCP timers don't work well. In *Proc. ACM SIGCOMM86*, pages 397–495, August 1986.

[Zha95] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83(10), October 1995.

Appendix A

Acronyms

AAL ATM Adaptation Layer

ABR Available Bit Rate

ABT ATM Block Transfer

ABT/DT ATM Block Transfer with Delayed Transmission

ABT/IT ATM Block Transfer with Immediate Transmission

ADSL Asymmetrical Digital Subscriber Line

ADTF Allowed Cell Rate Decrease Time Factor

ANSI American National Standards Institute

ARP Address Resolution Protocol

ATM Asynchronous Transfer Mode

ATMARP ATM Address Resolution Protocol

ATMSAP ATM Service Access Point

B-ICI Broadband Inter-Carrier Interface

B-ISDN Broadband Integrated Services Digital Network

BSD Berkeley Software Distribution

BUS Broadcast and Unknown Server

CAC Connection Admission Control

CATV Community Antenna Television

CBR Constant Bit Rate

CDF Cutoff Decrease Factor

CDV Cell Delay Variation

CDVT Cell Delay Variation Tolerance

CI Congestion Indication

CL Connectionless

CLP Cell Loss Priority

CLR Cell Loss Ratio

CNET Center National d'Etude des Télécommunications

CO Connection-Oriented

CPCS Common Part Convergence Sublayer

CRC Cyclical Redundancy Check

CS Convergence Sublayer

CT Controlled Transfer

DBR Deterministic Bit Rate

DFBA Differential Fair Buffer Allocation

DGCRA Dynamic Generic Cell Rate Algorithm

DMA Direct Memory Access

DSL Digital Subscriber Line

EFCI Explicit Forward Congestion Indication

EPD Early Packet Discard

EPFL Ecole Polytechnique Fédérale de Lausanne

ER Explicit Rate

FDDI Fiber Distributed Data Interface

F-GCRA Frame-Generic Cell Rate Algorithm

FIFO Firt-In First-Out

FRTT Fixed Round-Trip Time

FTP File Transfer Protocol

GCRA Generic Cell Rate Algorithm

GFC Generic Flow Control

GFR	Guaranteed Frame Rate
HBO	High Buffer Occupancy
HEC	Header Error Check
HFC	Hybrid-Fiber Coax
ICR	Initial Cell Rate
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
ILMI	Interim Layer Management Interface
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ISO	International Standardization Organization
ISP	Internet Service Provider
ITU-T	International Telecommunications Union - Telecommunications sector
LAN	Local Area Network
LANE	LAN Emulation
LBO	Low Buffer Occupancy
LECS	LAN Emulation Configuration Server
LES	LAN Emulation Server
LIS	Logical IP Subnet
LLC	Logical Link Control
MAC	Medium Access Control
MAN	Metropolitan Area Network
MBS	Maximum Burst Size
maxCTD	Maximum Cell Transfer Delay
MCR	Minimum Cell Rate
MFS	Maximum Frame Size
MIB	Management Information Base
MPEG	Motion Pictures Expert Group

MPLS Multi-Protocol Label Swapping

MPOA Multi-Protocol over ATM

MPS MPOA Server

MSS Maximum Segment Size

MTU Maximum Transmission Unit

NHRP Next Hop Resolution Protocol

NFS Network File System

NI No Increase

N-ISDN Narrowband Integrated Services Digital Network

NNI Network-Network Interface

nrt-VBR non realtime Variable Bit Rate

NSAP Network Service Access Point

OAM Operation and Maintenance

OSI Open Systems Interconnection

PBS Partial Buffer Sharing

PCM Pulse Code Modulation

PCR Peak Cell Rate

PDUs Protocol Data Units

PMD Physical Media Dependent sublayer

P-NNI Private Network-Network Interface

PNO Public Network Operator

PPD Partial Packet Discard

ppCDV Peak-to-peak Cell Delay Variation

PTI Payload Type Indicator

PVC Permanent Virtual Channel connection

QFC Quantum Flow Control

QoS Quality of Service

RDF Rate Decrease Factor

RED Random Early Detection

RIF Rate Increase Factor

RIO Random Early Detection with In/Out

RM Resource Management

RTT Round-Trip Time

rt-VBR realtime Variable Bit Rate

SAR Segmentation and Reassembly

SACK Selective Acknowledgement

SBR Statistical Bit Rate

SCFQ Self-Clocked Fair Queueing

SCR Sustainable Cell Rate

SDH Synchronous Digital Hierarchy

SDUs Service Data Units

SNAP SubNetwork Access Point

SSCF Service Specific Coordination Function

SSCS Service Specific Convergence Sublayer

SSCOP Service Specific Connection Oriented Protocol

SVC Switched Virtual Channel connection

TAT Theoretical Arrival Time

TBA Token-based Buffer Allocation algorithm

TBE Transient Buffer Exposure

TC Transmission Convergence

TCP Transmission Control Protocol

TDM Time Division Multiplexing

TOS Type of Service

TRM Time between RM cells

TSDU Transport Service Data Unit

UBR Unspecified Bit Rate

UBRw Weighted Unspecified Bit Rate

UDP User Datagram Protocol

UNI User Network Interface

UPC Usage Parameter Control

UU User to User indication

VBR Variable Bit Rate

VC Virtual Channel

VCC Virtual Channel Connection

VCI Virtual Channel Identification

VD Virtual Destination

VP Virtual Path

VPC Virtual Path Connection

VPI Virtual Path Identification

VPN Virtual Private Network

VS Virtual Source

VS/VD Virtual Source/Virtual Destination

WAN Wide Area Network

WDM Wavelength Division Multiplexing

WFBA Weighted Fair Buffer Allocation

WFQ Weighted Fair Queueing

WRR Weighted Round-Robin

WWW World WIde Web

xDSL Generic name for Digital Subscriber Line technologies (e.g. ADSL, VDSL, ...)

XTP eXpress Transport Protocol

XTPX eXpress Transport Protocol eXtended

Appendix B

The STCP simulator

The simulations described in this thesis were performed with the STCP simulator developed by Sam Manthorpe at EPFL [Man96]. STCP is a custom event-driven simulator which has been written to study the behavior of TCP in ATM networks. STCP provides models of queues, links, leaky buckets, background sources, ...An important feature of STCP is its speed since STCP was specifically written and tuned to simulate TCP over ATM. We describe in section B.1 the main components of STCP and briefly discuss our modifications to STCP in section B.2 Additional details on STCP and its features may be found in [Man96] and the STCP source code.

B.1 The main components of STCP

The main components of STCP that we used for this thesis are the workstations which generate TCP traffic, the queues which are used to model ATM switches and the ABR adapters and switches.

B.1.1 Workstations

The main characteristic of STCP compared with other simulators is that STCP does not contain a simplified model of TCP which is used for the simulations. Instead, STCP contains the real 4.4 BSD Lite [WS95] working TCP code. This TCP implementation supports all the important TCP mechanisms (slow-start and congestion avoidance, fast retransmit and fast recovery, Nagle algorithm, delayed acknowledgements, large windows and timestamp options[Ste94], ...). Furthermore, STCP emulates the socket layer, and thus the interactions between TCP and the application using it are also taken into account.

In STCP, the workstations always appear in pair according to the client-server model. The source workstation sends a file to the destination workstation. A source workstation always sends traffic to the same destination workstation. These file transfers are on-off file transfers. However, it should be noted that the on-off file transfers used by STCP are different from the used on-off sources. In STCP, the source workstation is idle during the off period. The transition from the off state to the on state corresponds to the beginning of the transmission of a file. A key characteristic of STCP is that the duration of the on period is equal to the time needed to transfer the file. Thus, the duration of the on period is a function of the status of the network. The on period will be much longer when the network is congested than when the network is not congested. The transition from the on state to the off state corresponds to the end of the file transmission. The characteristics of these on-off sources are controlled by specifying the mean value of two exponentially distributed random variables : the duration of the off period and the file length.

B.1.2 Queues

The queues are used to model a FIFO buffer attached to an output link. An interesting feature of the queues used by STCP is that they support several buffer acceptance algorithms. The original STCP simulator supported three algorithms : tail-drop, PPD and EPD. However, due to the modular nature of the STCP source code, it is possible to support other buffer acceptance algorithms by writing a few C functions.

B.1.3 ABR adapters and switches

STCP supports the ABR service category as defined in [For96c]. The ABR adapters are not included inside the workstations, but are instead attached to independent queues. In addition to the ABR adapters, STCP also supports the ERICA and ERICA+ switch algorithms.

B.1.4 Modeling an ATM switch

Throughout this thesis, we have always considered non-blocking ATM switches. An example of such switches is the bus-based output-buffered ASX-200 BX switch that we used for the measurements in chapters 4 and 6. The ASX-200 BX is composed of several ports which are all attached to a high-speed bus. Each port contains one input queue and one output queue for each physical link it supports. A good model of the switch would require one STCP queue for each input queue and one STCP queue for each output queue and a model of the bus. However, in this switch, very few queueing occurs on the input queues since the bandwidth on the bus is larger than the sum of the bandwidths of the links attached to the switch. Thus, most of the queueing occurs in the output queues. For this reason, we did not include the input queues and the bus in the ATM switches used in our simulation models.

B.2 Our modifications to STCP

We explain in this section the main functions we added to STCP in order to perform the simulations described in this thesis.

B.2.1 Modifications to the workstations

We have slightly modified the workstations defined in STCP. We have added several trace functions which were very useful to understand some of the simulation results. We have also patched the TCP code in STCP to support the Selective Acknowledgement (SACK) option. For this, we have used the NetBSD SACK implementation written by Mahdavi [Mah96]. This implementation is rather conservative in its handling of retransmissions and follows the guidelines proposed in [FF96].

B.2.2 Modeling a router

The original STCP simulator only contained queues which could be used to model ATM switches. We have modified STCP to include a simple model of a router with an ATM interface. Our model of a router utilizes the VC-merging approach. It is composed of two queues as shown in figure B.1. Our main goal when developing this model was to avoid the interleaving of cells from different AAL5-PDUs on the output link of the routers. STCP only supports ATM cells and modifying STCP to support variable length packets and ATM cells in the same simulation would have been a major change. For this reason, we continued to use ATM cells on all the links in our simulation. The reassembly queue of the router is used to temporarily store the partially received AAL5-PDUs. When the last cell of one AAL5-PDU is received in the reassembly buffer, the complete AAL5-PDU is transmitted in zero time to the output queue. The output queue sends its ATM cells on the output link of the router. A similar model of a router was also used in [GJF+97].

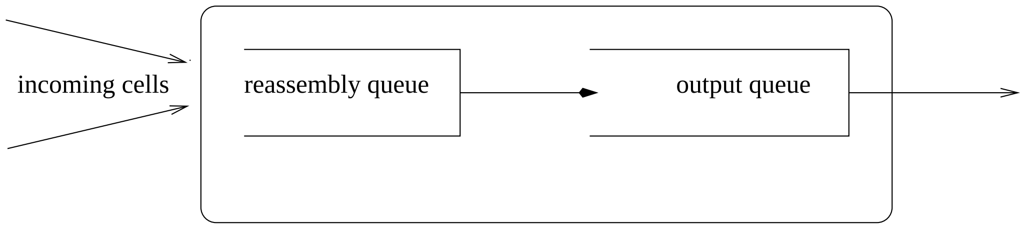

Figure B.1: Model of a router

Our model of a router has some limitations. The first limitation is that it receives ATM cells and transmit ATM cells instead of IP packets. This was not a problem for our simulations with internetwork traffic described in chapters 6 and 7 since in the scenario we studied the workstations were directly attached to the routers which were attached to ATM backbones. The second limitation is that it does not verify the CRC of the incoming AAL5-PDU. Our router could thus transmit corrupted AAL5-PDUs, in contrast with a real router. This limitation was not a problem for our GFR simulations since the GFR switches only discard complete AAL5-PDUs. For the VBR.3 simulations with internetwork traffic, the corrupted AAL5-PDUs were detected by the destination workstations instead of the destination routers.

B.2.3 F-GCRA

The F-GCRA that we have added to STCP is a straightforward extension of the CBR leaky buckets which were already supported by STCP. Our F-GCRA is implemented as a control function which can be attached to any queue as the CBR leaky buckets.

B.2.4 GFR switch implementations

We have modified STCP to support the three GFR switch implementations studied in chapters 7 and 8.

Double-EPD

The Double-EPD buffer acceptance algorithm has been implemented as a packet discard function in STCP. It can be used with any queue. The Double-EPD buffer acceptance algorithm has been implemented like the EPD buffer acceptance algorithm which was already supported by STCP.

Per-VC threshold and scheduling

The original STCP simulator did not contain any WFQ-like scheduler. However, STCP supported priority queues. These priority queues are served with a strict priority scheduler (queue[0] has a strict priority over queue[1] which has a strict priority over queue[2] ...). We have reused a part of the code used to implement the priority queues for the per-VC threshold and scheduling implementation and have replaced the strict priority scheduler by the Virtual Spacing algorithm. In our model of the per-VC threshold and scheduling implementation, a MCR and a T_i threshold are associated with each logical queue. Our model currently only supports the buffer acceptance algorithm defined in [GH96], but the code is modular and other buffer acceptance algorithms could be easily added.

```
Cell transmission from GFR buffer:

  j=(current_pos+1) mod $W_R$;
  while not( filled($W_R$[j]) and C[$W_R$[j]] < MBS[$W_R$[j]] )
    { /* find active VC */
      j=(j+1) mod $W_R$;
    }
  C[$W_R$[j]]=C[$W_R$[j]]+1; /*incr.  counter of found active VC*/
  current_pos=j;
```

Figure B.2: Counter update algorithm implemented in the simulator

TBA switch implementation

We have modified STCP to support the main features of the Token-based Buffer Allocation algorithm (TBA) proposed in section 8.3. Our model implements the buffer acceptance algorithm described in figure 8.16 on page 159 and a simplified counter update algorithm. Instead of implementing the complete counter update algorithm shown in figure 8.15 on page 156, we have implemented a simpler counter update algorithm [Bon98a]. This counter update algorithm (figure B.2) relies only on the W_R register and distributes the excess tokens in proportion to the MCR of the active VCs. This counter update algorithm is equivalent to the complete counter algorithm of figure 8.15 provided that the W_R and W_U registers have the same size and that the administrative weights are equal to the MCR of the GFR VCs.

Appendix C

A brief overview of TCP

In this appendix, we briefly describe the main characteristics of the data transfer mechanisms used by the Transmission Control Protocol (TCP) and their evolution since the first TCP specification [Pos81b]. A more detailed description of TCP may be found in books such as [Ste94].

This appendix is structured as follows. We first briefly describe the main characteristics of the original TCP specification. We then discuss the five major parts of the TCP protocol, namely the packet transmission and the acknowledgment strategies, the various retransmission mechanisms, the methods used to measure the round-trip-time and the congestion control mechanisms.

C.1 The original TCP specification

The Transmission Control Protocol (TCP) provides a reliable byte-stream service over the unreliable connectionless IP service. The original TCP specification [Pos81b], which appeared in the early eighties, is still the reference for the packet[1] formats (figure C.1), but most of the mechanisms associated with the actual transmission of data have been modified since then.

The main fields of the TCP header are the source and destination port numbers, the flags, the sequence and acknowledgment numbers, the window size and the checksum. The source and destination port numbers are used together with the source and destination IP addresses to unambiguously identify a TCP connection. Each TCP connection is identified by the four-tuple <*source IP address, destination IP address, source port, destination port*>. The flags are used during the connection establishment and release phases to perform the three way handshake. During the data transfer phase, the ACK flag is used to indicate whether the packet contains a valid acknowledgment number.

TCP relies on a sliding window protocol to provide a reliable byte-stream service. Each TCP segment is protected by a 16-bits long checksum. Furthermore, each TCP segment contains a 32-bits long sequence number which corresponds to the sequence number of the first user byte of the segment. The acknowledgments are also encoded in 32-bits fields. The main limitation of the original TCP specification was probably the decision

[1]In the Internet community, a TCP packet (or PDU) is called a "segment". We adopt this terminology in the remainder of this appendix.

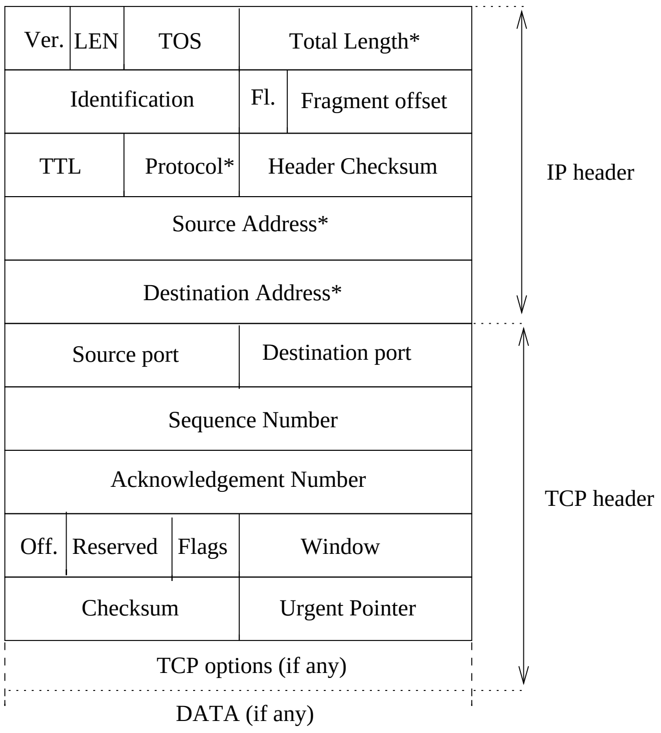

Figure C.1: The IP and TCP headers

to encode the window size into a 16 bit field. Due to this choice, the maximum window size supported by many TCP implementations is limited to 65535 bytes. Such a small window size is a severe limitation in high-speed networks.

Fortunately, the original TCP specification allowed the TCP header to be extended with optional fields. In [Pos81b], these options were only used to negotiate the maximum size of the TCP segments (Maximum Segment Size (MSS)) at connection establishment time. Although this option can be used to negotiate different MSS values for the two directions of data transfer, most TCP implementations use the minimum MSS value announced during the three way handshake for both directions of data transfer.

Another important option is the "window scale factor" option proposed in [JBB92]. With this option, RFC1323-compliant TCP implementations can utilize window sizes much larger than 65535 bytes. Instead of modifying the TCP header, [JBB92] choose to require TCP implementations to store the window size internally as a 32-bits number and to use the "window scale factor" option during connection establishment to negotiate the scaling factor for the window field of the TCP header. As the 32-bits window cannot be transmitted in the TCP header, the window field of the TCP header is the used to transmit the 32-bits window right shifted by the "window scale factor". The "window scale factor" can be any number of bits between 0 and 14.

The original TCP specification defined in [Pos81b] has benefited from the experience of using TCP in the Internet. This operational experience allowed the TCP users and developers to improve the performance of the protocol in various situations. Since 1981, a constant flow of improvements and modifications have been proposed for TCP, mainly for the data transfer phase. While most of these modifications have always maintained the backward compatibility with the 1981 TCP specification, it is clear that the TCP implementations used today have large differences with the early TCP implementations. Since the TCP header has not changed since 1981, an early TCP implementation would probably inter-operate with a current TCP implementation, but the performance would probably be quite low.

C.2 The segment transmission strategies

The original TCP specification did not explicitly specify how and when a TCP segment should be transmitted. This issue was left at the discretion of the implementors, but after some operational experience it appeared that some performance problems occurred.

C.2.1 The Nagle algorithm

After some operational experience with TCP, it appeared that some implementations were sometimes sending one TCP segment for each byte sent by the application. In a LAN, this behaviour was not a real problem, but in a WAN, this caused a huge overhead as 40 bytes of IP and TCP headers were added to each user byte. A solution to reduce the overhead due to the transmission of small packets, while still allowing short response times was proposed in [Nag84]. This algorithm, known as the Nagle algorithm, specifies that a TCP implementation is only allowed to send a small segment (i.e. a segment shorter than the negotiated MSS size) when all the previously sent bytes on this connection have been acknowledged. Otherwise, it shall only send MSS-sized segments. The Nagle algorithm must be supported by all TCP implementations [Bra89], but there must be a mechanism to disable it (XWindow, for example, usually disables the Nagle algorithm on the TCP connection it uses). The Nagle algorithm is usually enabled by default.

C.2.2 The Silly Window Syndrome

The silly window syndrome [Cla82] is a pathological condition that may occur with the sliding window mechanism used by TCP. When it occurs, small segments are exchanged instead of full-size segments. It may be caused by either the sending end if it sends small segments, or by the receiving end if it advertises small windows. To avoid this pathological condition, TCP uses the following algorithm [Cla82].

- A receiver shall not advertise a window that is not at least equal to the negotiated MSS, or one half of the total receive window.
- A sender shall not send a segment unless either :
    1. The segment is a full-sized segment
    2. The segment corresponds to at least 50% of the maximum window size ever advertised by the receiver on this connection
    3. The segment empties the send buffer and there are no unacknowledged bytes
    4. The Nagle algorithm is disabled for this connection

C.3 The acknowledgment strategies

When discussing the TCP acknowledgment strategies, we have to consider the acknowledgments supported in the original TCP specification and the recently proposed selective acknowledgments.

C.3.1 The delayed acknowledgments

According to the original TCP specification [Pos81b], the ACK field of the TCP header was used to carry the sequence number of the last byte received in-sequence. However, this specification did not provide strict guidelines on how and when to send segments containing only acknowledgments. Several acknowledgment strategies can be used, at least in theory, with TCP. The simplest one is to send an acknowledgment as soon as a segment is received. The advantage of this strategy is that this gives a clear view of the progress of the data transfer to the sender. However, as each acknowledgment requires some processing on both the sender and the receiver, and also requires some amount of bandwidth, this may not be the optimal acknowledgment strategy in all environments.

The acknowledgment strategy used by most TCP implementations and required by [Bra89] was proposed in [Cla82]. This acknowledgment strategy relies on delayed acknowledgments. When a TCP entity receives a data segment, it shall not immediately send an acknowledgment for this segment. Instead, it shall notice that an acknowledgment is due, and start its "delayed acknowledgment" timer. An acknowledgment shall be piggybacked with a new data segment if such a data segment is sent and the timer shall be deactivated. If no data segment is sent, an acknowledgment shall be sent if a second data segment is received, or upon the expiration of the "delayed acknowledgment" timer. It is recommended [Bra89] that the "delayed acknowledgment" timer be shorter than 500 milliseconds and most TCP implementations use either 200 or 50 milliseconds [Ste94]. With this delayed acknowledgment strategy, a TCP implementation acknowledges thus every second segment during a bulk data transfer.

C.3.2 The selective acknowledgments

Although the acknowledgment strategy described in the previous paragraph is still the only strategy supported by most TCP implementations, it is far from perfect. Relying exclusively on positive and cumulative acknowledgments is not the best solution from a performance point of view. A new acknowledgment strategy, based on selective and positive acknowledgments was recently accepted[2] by the IETF [MMFR96]. It can be expected that this new acknowledgment strategy will be supported by the major TCP implementations.

This new acknowledgment strategy relies on the SACK options which are used as follows. The utilization of the SACK is negotiated during connection establishment. Each "SACK" option indicates a received block of data, and contains two 32-bits fields. The first one is the first sequence number of the block, and the second is the sequence number that immediately follows the last sequence number of the block. As the TCP header has

[2]A similar option has already been proposed in [JB88], but it was rejected by the IETF at that time.

a maximum length of 64 bytes, there may be up to 4 SACK blocks within a single TCP header. The following rules apply when generating SACK blocks [MMFR96] :

- The first SACK block must correspond to the last received segment, unless this segment advanced the acknowledgment field in the header
- As many distinct SACK as possible shall be included in the TCP header
- The SACK option should be filled by repeating the most recently reported SACK blocks. This rule is used to ensure that each SACK block will be repeated at least three times when the timestamp options (see section C.5) are used.

C.4 The retransmission mechanisms

The retransmission mechanisms used in the TCP implementations have evolved since the first TCP implementations. Although the first TCP implementations exclusively relied on one retransmission timer to perform retransmissions, recent implementations support other retransmission mechanisms.

C.4.1 The default timer-based retransmissions

By imposing positive cumulative acknowledgments, the original TCP specification has also imposed the GO-BACK-N retransmission strategy with retransmission upon expiration of the retransmission timeout. This timer-based retransmission mechanism is the default retransmission mechanism supported by all TCP implementations.

C.4.2 The fast retransmit algorithm

While the default retransmission strategy is sufficient to recover from losses, its performance is not optimal. The fast retransmit algorithm [Ste97] tries to avoid some retransmission timeouts by requiring TCP endsystems to generate an acknowledgment when they receive out-of-sequence segments. When a TCP endsystem receives d (usually 3) duplicate acknowledgments, it assumes that the requested segment has been lost, and this segment is retransmitted selectively. This selective retransmission heuristic helps to avoid costly expirations of the retransmission timeout, but has difficulties when several segments are lost inside a single window. In some cases, the fast retransmit and recovery algorithm may have a lower performance than the default (timeout based) retransmission algorithm [FF96].

Other improved retransmission mechanisms have been proposed in the literature, but they are not currently used by major TCP implementations. [BOP94] proposed an elegant heuristic to perform selective retransmissions without requiring the SACK option. [Hoe96] proposed an improvement to the fast retransmit algorithm which is able to better recover from multiple packet losses in the same window.

C.4.3 Retransmissions with the selective acknowledgments

There is currently no recommended retransmission strategy for a SACK-compliant TCP. In order to deal with reordered segments in an Internet, a common solution will probably be to retransmit missing segments when at least d (for example 3) SACK blocks have indicated that these segments were missing. Other solutions are possible.

The SACK-based retransmission mechanism will have to take the congestion control mechanisms (see below) into account. [MMFR96] recommends that a SACK-based TCP implementation still use the slow-start, congestion avoidance and fast recovery mechanisms. Furthermore, when retransmissions occur, a sender shall limit the number of segments that are sent in response to each ACK segment to one or two segments. Other retransmission strategies suitable for a SACK-compliant TCP are discussed in [MM96].

C.5 Estimating round-trip-time

As the main retransmission mechanism in TCP is go-back-n triggered by a retransmission timer, an accurate evaluation of the round-trip-time (rtt) is important. If the rtt estimation is too conservative (i.e. the estimated rtt is much larger than the actual value), unnecessary idle times will occur when segments need to be retransmitted. On the other hand, if the rtt estimation is too optimistic, unnecessary retransmissions will occur.

The original TCP specification [Pos81b] recommended to estimate the round-trip-time with a weighed average and to set the retransmission timer to twice the estimated rtt. This rtt estimation was the source of several performance problems [Zha86]. The main problems were that this rtt estimation did not adapt quickly to changes in the round-trip-time due to changing network conditions and that it did not consider retransmissions. These problems were addressed by two distinct improvements.

The first improvement was proposed in [KP87]. To avoid problems when retransmission occurs, [KP87] specified that the rtt should not be measured on a retransmitted segment. This is due to the fact that when a TCP sender receives an acknowledgment for a retransmitted packet, it cannot determine whether the acknowledgment corresponds to the reception of the original transmission or to the reception of the retransmission. While the rtt estimates cannot be updated with retransmitted segments, [KP87] requires that a TCP implementation applies a back-off strategy to the retransmission timer value when this timer expires, and that the backed-off value be maintained until a valid rtt sample is obtained. The suggested back-off strategy is to double the value of the retransmission timeout upon its expiration.

The second improvement was to take into account the variance of the round-trip-time measurements in the retransmission timeout. An elegant way to perform this estimation was proposed in [Jac88]. This algorithm is used by most TCP implementations.

In [JBB92] a new TCP option was proposed to improve the estimation of the round-trip-time. This option may appear in any TCP segment. Before the introduction of this option, most TCP implementations measured the rtt for a single segment per window. With large windows, such a low sampling interval may produce incorrect round-trip-time estimations. The solution proposed in [JBB92] was to include two "timestamps" ("timestamp value" and "timestamp echo reply" inside each segment with the timestamp option. The "timestamp value" is inserted by the TCP entity which sends the segment. When a TCP entity generates an acknowledgment, it stores in the "timestamp echo reply" the timestamp received in the earliest segment. When a TCP sender receives an acknowledgement, it can easily use the "timestamp echo reply" to determine the rtt of the acknowledged segment. While it could have been possible to measure the rtt more frequently without using the timestamp options (see for example [BOP94]), the timestamp options are also useful to protect against wrapped sequence numbers in a network that may reorder segments.

C.6 The congestion control mechanisms

In the original TCP/IP specifications [Pos81a, Pos81b], the congestion problem was only briefly considered. When the Internet suffered from periods of congestion collapse in the late 1980's, Van Jacobson proposed [Jac88] and implemented a congestion control mechanism in TCP that relies on segment losses as the sole indication of congestion. This mechanism is supported by most TCP implementations.

C.6.1 The slow-start and congestion avoidance mechanisms

The congestion control mechanism proposed in [Jac88] is divided in two phases : the slow-start and the congestion avoidance. These two algorithms adapt the "rate" of transmission of TCP segments by constraining the instantaneous window size used by the sending TCP entity. They both rely on two additional state variables per TCP connection: `cwnd` (the congestion window) and `sstresh` (the slow-start threshold). The congestion window is used to artificially limit the window used by the sender to *minimum(*`cwnd`*,window advertised by the receiver)*. The slow-start threshold is used to switch from the slow-start phase to the congestion avoidance phase.

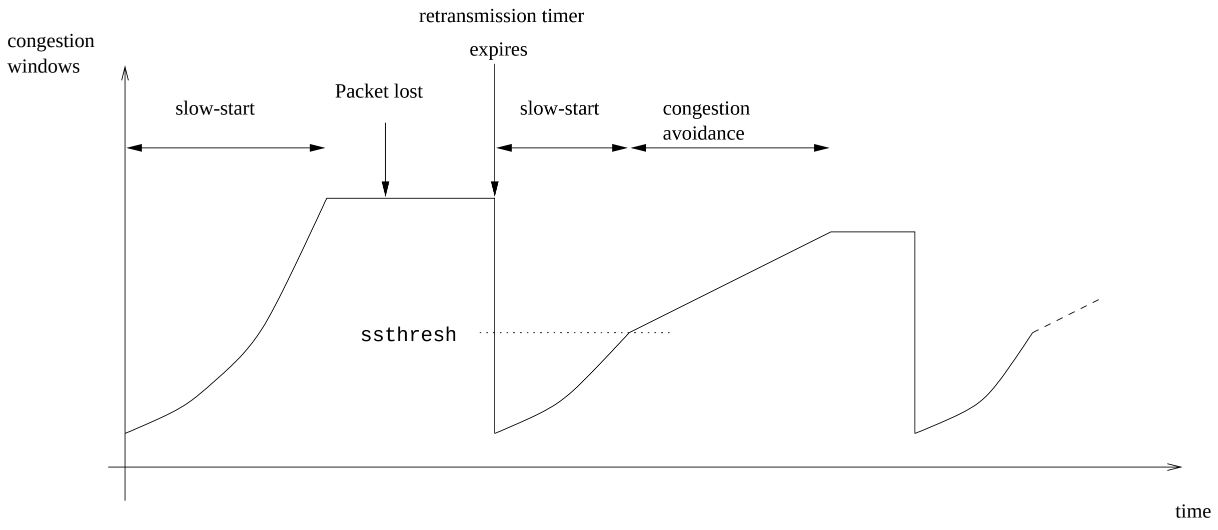

Figure C.2: The slow-start and congestion avoidance mechanisms

The slow-start mechanism is used when a TCP connection starts or has been idle for a time larger than the retransmission timeout. Initially, the congestion window is set to one MSS-sized segment, and the slow-start threshold is set to the maximum window size of the sender. This allows the sender to send a MSS-sized TCP segment. During slow-start, each time the sender receives an acknowledgment, it increases the value of its congestion window by one segment. Thus, TCP effectively doubles its congestion window after each round-trip-time during slow-start (figure C.2). The slow-start phase ends when the congestion window reaches the slow-start threshold.

During the congestion avoidance phase, TCP increases its congestion window by one segment per round-trip-time. As the transmission rate is proportional to the value of `cwnd` divided by the rtt, the transmission rate is increased exponentially during slow-start and linearly during congestion avoidance. When a segment is retransmitted due to the expiration of the retransmission timeout, TCP assumes that the loss was caused by some congestion and the slow-start threshold is set to half the current value of the congestion window and the congestion window is reset to one full-sized segment and TCP performs slow-start. The slow-start and congestion avoidance mechanisms are summarized in figure C.2. Other types of congestion control algorithms have been proposed [BOP94, WC91, WC92, MM96], but they are either not implemented, or only supported by experimental implementations and thus are not be considered in this thesis.

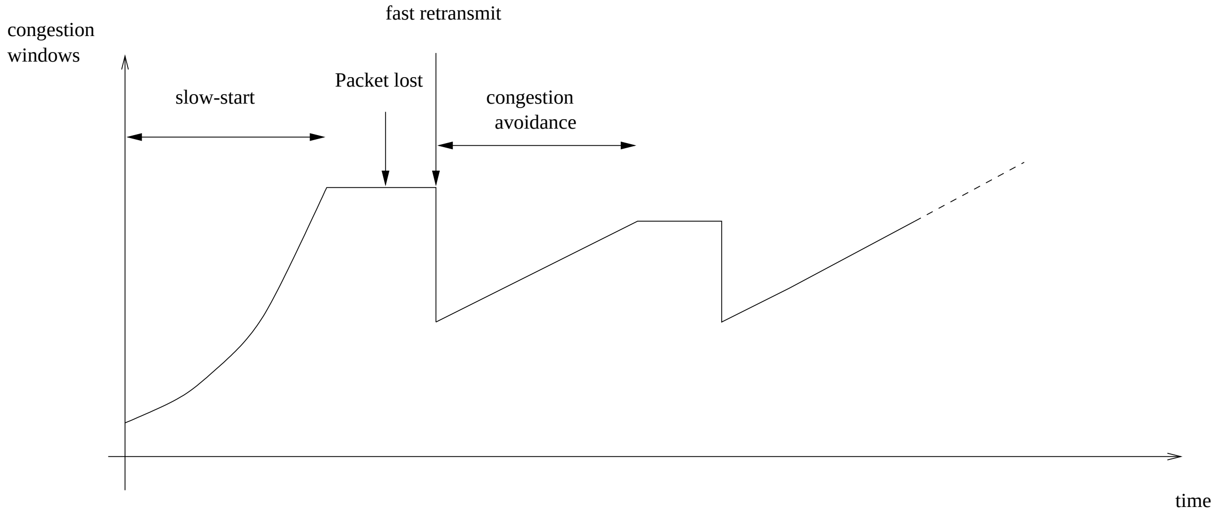

Figure C.3: The fast recovery mechanism

C.6.2 The fast recovery mechanism

The fast recovery algorithm is an extension of the slow-start and congestion avoidance mechanism. With the original TCP congestion control mechanism, the TCP congestion window was reset to one segment after each segment loss. As the sender was forced to perform slow-start after each segment loss, this caused some performance degradations. The fast recovery algorithm tries to reduce these performance degradations, while still preserving the additive increase and multiplicative decrease characteristic of TCP's congestion control algorithm. This is done by going directly into the congestion avoidance phase when a segment is retransmitted by the fast retransmit algorithm (i.e. both the congestion window and the slow-start threshold are set to half the current value of the congestion window). A detailed description of the interactions between fast retransmit and fast recovery may be found in [Ste97].

List of Tables

List of Figures